

Temas-FR.pdf



Evanpheus



Fundamentos de Redes



3º Grado en Ingeniería Informática



**Escuela Técnica Superior de Ingenierías Informática y de
Telecomunicación
Universidad de Granada**

Estudiar sin publi es posible.



Compra Wuolah Coins y que nada te distraiga durante el estudio



Descubre la gama ZBook para creativos.

En Septiembre con 6% dto. adicional sólo para ti. Usa el cupón: WUOLAH6

Últimos días del cupón



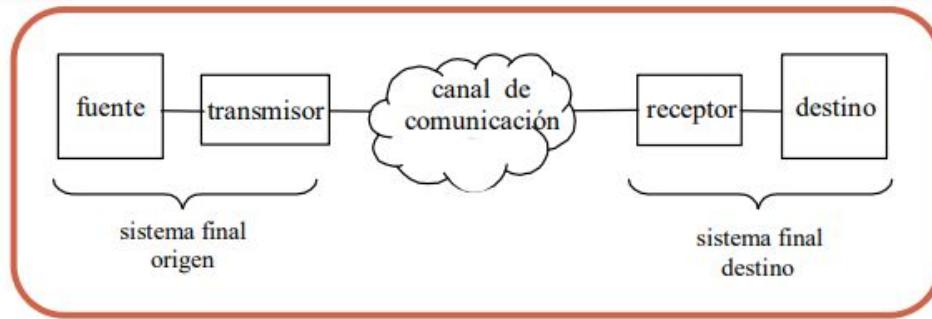
Tema 1

Introducción a los Fundamentos de Redes

1.1 Sistemas de Comunicación y Redes

Sistemas de Comunicación: Conjunto de elementos y dispositivos involucrados en la transmisión de información entre dos puntos remotos. Elementos:

- **Fuente o emisor:** origen de la información.
- **Destino:** destinatario último de la información transmitida.
- **Canal o medio de transmisión:** soporte físico sobre el que se realiza el transporte de la información.
- **Transmisor:** dispositivo interfaz entre la fuente y el canal, encargado de llevar a cabo la adaptación de la señal a transmitir a las características del medio de transmisión.
- **Receptor:** dispositivo interfaz entre el destino y el canal, encargado de llevar a cabo la adaptación de la señal recibida sobre el canal a las características del destino.



Información: Conjunto de datos que se dota de significado. Diferenciamos entre el contenido (información como tal) y continente (señal que sirve de soporte a la misma).

Redes de computadores: sistema de comunicación con sistemas finales (terminales) autónomos (con capacidad de procesar información) que facilita el intercambio eficaz (maximizar los recursos) y transparente (la información fluye superando las heterogeneidades de manera autónoma) de la información. Dos principales características:

- **Interconexión:** Existen uno o varios canales de transmisión entre los equipos que forman la red
- **Autonomía:** Los equipos son independientes entre sí en cuanto a funcionamiento, diferencia una red de ordenadores del concepto de mainframes (varios equipos terminales dependientes de una única unidad central). También distingue redes de computadores de sistemas distribuidos (integración conseguida por el SO es tal que el uso de los distintos computadores y recursos resulta transparente para el usuario).



WUOLAH

Estructuras de las Redes

Nodo: Cada uno de los equipos que componen una red y que presentan capacidades de comunicación.

Según el modo de utilización del canal de comunicación, podemos formar dos tipos de interconexiones:

- a. **Difusión:** Un único medio de transmisión compartido por todos los equipos, sobre el que se realizan todas las transmisiones, se les suele llamar multipunto. (WiFi, Bluetooth...)
- b. **Punto a punto:** Intervienen solo dos equipos utilizando un canal (cable) al que solo ellos tienen acceso. (Fibra, ADSL...)

La difusión tiene problemas de escalabilidad y resulta inviable la existencia de un número alto de equipos o una extensión geográfica relativamente amplia de la misma (problemas de retardos). Solo un canal de comunicación compartida, solo se puede producir una comunicación en cada instante de tiempo.

Se vieron necesarias unas conexiones punto a punto, lo malo es que tienen que estar todos los equipos conectados y permitir conexiones simultáneas y esto es muy costoso. Por eso se recurre a una nueva estructura, centrales de conmutación (llegan hilos correspondientes a los terminales de los usuarios, se establecen conexiones físicas entre los pares de hilos de los usuarios que deseaban establecer comunicación). Surgen así las estructuras jerárquicas, donde podemos diferenciar entre dos tipos de comunicaciones:

Comunicaciones salto a salto: Tienen lugar de forma directa entre cada par de nodos involucrados en la comunicación entre dos nodos terminales.

Comunicaciones extremo a extremo: Entre nodos terminales. Normalmente implican la participación de nodos intermediarios.

Hosts: Nodos o equipos terminales de datos, también conocidos como sistemas finales o estaciones de trabajo. Ej: móvil, tablets...

Subred: Conjunto de elementos que posibilitan la interconexión con los hosts. Compuesta por:

- **Nodos de conmutación:** Dispositivos que posibilitan el transporte de datos entre un origen y un destino. Ej: Router
- **Nodos de transmisión:** Canales o enlaces de comunicación. Ej: wifi, fibra ADSL...

Por escala o multipunto(en función de su extensión o cobertura geográfica):

- **LAN:** Local Area Network, Red de área local, abarca centenas de metros incluso km, usa tecnología basada en difusión.
- **MAN:** Metropolitan Area Network, Red de área metropolitana, está en desuso.
- **WAN:** Wide Area Network, Red de área amplia, puede llegar a cubrir un país o un continente entero, usa tecnología basada en transmisión punto a punto y nodos de conmutación.
- **PAN:** Red de área personal. Alcance de pocos metros para interconectar los equipos de un usuario.

En la actualidad no existen diferencias tecnológicas significativas entre LAN y MAN.

1.2 Diseño y Estandarización de Redes

Problemas a resolver por la red para poder conseguir transparencia y eficacia:

- ¿Cómo enviar físicamente la información?
- Compartición del medio. ¿Quién accede? Segmentación de la información mediante tramas (transmitir varios bytes)
- Control de errores y control de flujo para no perder la información (también a nivel de tramas)
- Control de encaminamiento (enrutamiento). Encontrar la mejor ruta para llegar al destino
- Control de congestión. Limitaciones de las redes (ancho de banda por ejemplo). La memoria de los routers es finita (se puede bloquear) • Entrega ordenada de los mensajes
- Gestión del diálogo (mecanismo para regular la emisión y recepción)
- Representación interna (sintaxis) de los datos
- Significado (semántica) de los datos

Adoptamos un diseño por capas para solucionar estos problemas: Este hecho además permitirá un sistema más modular y consecuentemente, más flexible. Modelo de referencia: definición por capas y para cada una la funcionalidad que resuelve. Los principios de este diseño son:

- Funcionalidades distintas deben estar en capas distintas
- Minimizar el flujo de información entre capas IETF -> se adoptan acuerdos recogidos en un RFC

Modelo de Referencia OSI

Se define en 7 capas, de abajo a arriba:

- **Física:** se llevan a cabo funciones relacionadas con la transmisión de datos desde el punto de vista de la gestión de características eléctricas, mecánicas y funcionales para una adecuada transferencia de información sobre el canal. La finalidad de esta capa es el intercambio de bits entre dos equipos conectados mediante un canal de transmisión (estos introducen errores que pueden alterar la información transmitida).
- **Control del enlace de datos:** A partir del servicio proporcionado por la capa física, esta capa debe realizar 3 funciones con la finalidad de gestionar adecuadamente conjuntos de bits, agrupados en una unidad llamada trama.
 - a. **Delimitación de tramas:** para conocer el principio y el fin de cada bloque de datos y así permitir la sincronización emisor-receptor.
 - b. **Control de errores:** para conseguir que la información recibida se corresponda con la original emitida.
 - c. **Control de flujo:** para evitar que el emisor sature la memoria de almacenamiento temporal (buffer) debido a una velocidad u ocupación diferente de las dos partes.

El objetivo de esta capa es la entrega al receptor de forma fiable y fidedigna de todos los datos enviados por el emisor.

- **Red:** dos funciones principales;
 - a. **Encaminamiento:** establecer una ruta (secuencia de líneas y nodos de commutación en la subred) a seguir desde un origen hasta un destino dados.
 - b. **Control de gestión:** para evitar la saturación de la capacidad de la subred como consecuencia de un elevado tráfico.

Otro aspecto importante de esta capa es el relativo a las interconexiones de redes, se posibilita la transmisión de datos entre estaciones finales situadas en redes distintas, también se incluyen en esta capa las funcionalidades asociadas a la tarificación (evaluación de los consumos generados).

El objetivo de esta capa es hacer que las unidades de datos de este nivel, paquetes, sean dirigidos al destino a través de las diferentes subredes de la forma más eficiente posible.

- **Transporte:** Se lleva a cabo un control de flujo y de errores extremo a extremo, la capa de transporte ve la subred no como un conjunto de nodos y enlaces, lo ve como un solo ente sobre el que los hosts o equipos finales emisor y destino deben controlar que la transmisión se lleva a cabo con éxito.

Otra función importante es la relativa a la multiplexación de aplicaciones sobre una misma conexión red (posibilitar varias comunicaciones entre los mismos hosts)

La unidad de datos del nivel de transporte es el segmento, siendo la finalidad básica de esta capa la entrega fiable de los segmentos entre el origen y el destino, extremo a extremo.

- **Sesión:** Gestiona el diálogo o <>turno de palabra>> en una comunicación entre los hosts participantes.
- **Presentación:** Se aborda la representación de los datos que provienen de la capa superior, esta capa resuelve las heterogeneidades respecto de la diferente representación interna de la información en cada uno de los hosts extremos.
- **Aplicación:** Hace referencia a los distintos estados finales que se ofrecen al usuario: correo electrónico, transferencia de ficheros, etc. Esta capa trata la información como tal intercambiada entre los usuarios finales.

Lo malo de este modelo es que algunas de sus capas están vacías o desequilibradas respecto a otras por eso otros modelos tienen menos capas.

Descubre la gama ZBook para creativos.

En Septiembre con 6% dto. adicional sólo para ti. Usa el cupón: **WUOLAH6**

Últimos días del cupón

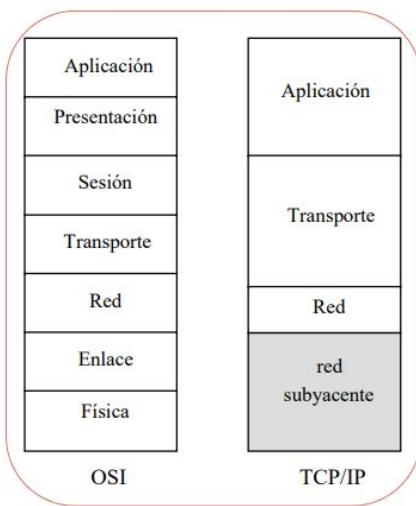


Modelo TCP/IP

Conocido dado su empleo en internet. El TCP/IP solo tiene 3 capas, de arriba a abajo:

- **Aplicación:** Incluye servicios de usuario (ssh, ftp, smtp...).
 - **Transporte:** Control de flujo, de errores, de congestión y de conexión (extremo a extremo).
 - **Red:** Ejecuta la función de encaminamiento y fragmentación.

Las tres capas mencionadas deben sustentarse sobre otras inferiores que permitan en última instancia la comunicación entre los sistemas finales. Aunque dentro del modelo TCP/IP se contemplan protocolos de capa inferior a la red, estos se refieren a la necesidad de integrar TCP/IP con la tecnología de red subyacente. Desde este punto de vista TCP/IP es una red software, de modo que puede implementarse sobre cualquier tecnología de red sin ser dependiente de ella. Esto es una de las razones por las que TCP/IP es el modelo dominante, convirtiéndolo en un ejemplo de estándar de facto.



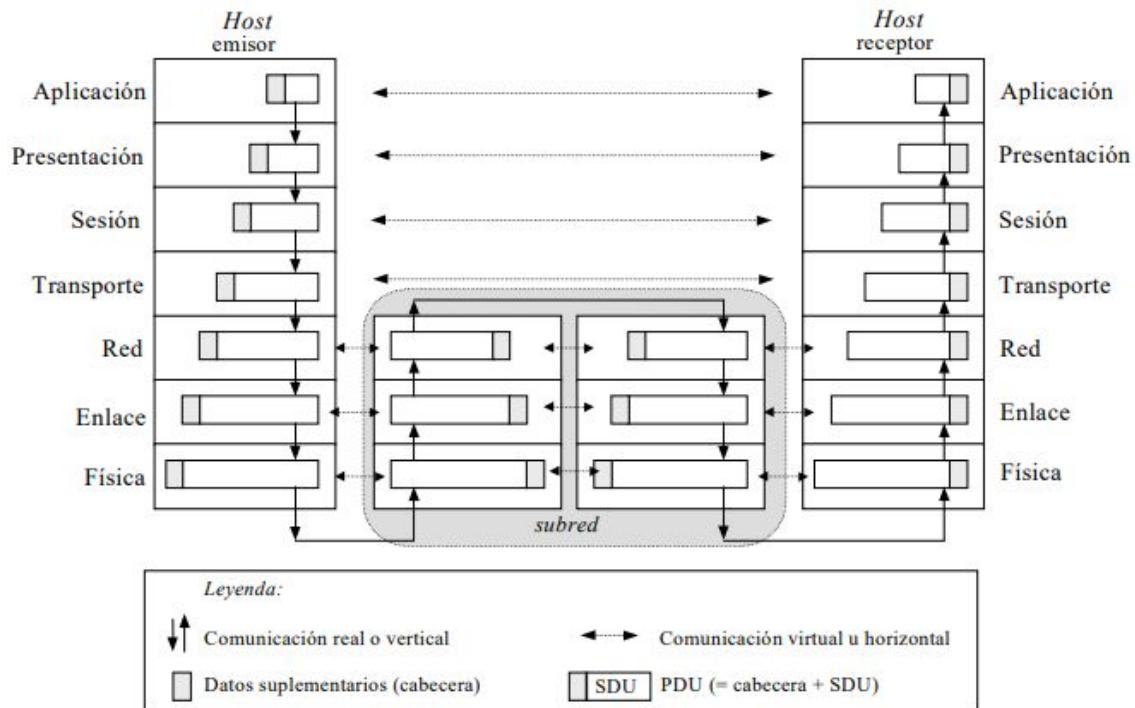
1.3 Terminología, Conceptos y Servicios

Los elementos activos, hardware o software, existentes en una capa N se conocen como entidades de nivel N. Por su parte, las entidades de nivel N en el emisor y en el receptor reciben el nombre de entidades pares o paritarias. A partir de este concepto ha de señalarse la existencia de dos tipos de comunicación entre un emisor y un receptor:

Comunicación real o vertical: Flujo que sigue la información entre el emisor y el receptor. Intercambio de datos entre capas adyacentes en sentido descendente en el emisor (de aplicación a física en OSI, por ejemplo) y en sentido ascendente en el receptor (de física a aplicación).

Comunicación virtual u horizontal: Comunicación observada desde el punto de vista de las entidades paritarias. En cada capa se añade una serie de información suplementaria en forma de cabecera que permite una comunicación coherente entre las entidades, esta

información sólo es relevante para dichas entidades estando asociada a funcionalidades o parámetros relativos a los servicios que debe proporcionar. Las cabeceras se irán eliminando al ir pasando los datos a las capas superiores, se produce así el denominado encapsulado de los datos (el bloque de datos de la capa N+1 es el que se incluyen las cabeceras, es tratado como bloque de datos sin significado ni estructura, en la capa N).



Carga útil (payload): Los datos transportados en cada bloque de datos.

Protocolo: Conjunto de reglas y convenciones a aplicar en una comunicación entre dos entidades paritarias con objeto de llevar a cabo una cierta función o servicio. Se basan en el paso de mensajes que desencadena determinadas actuaciones por parte de una de las entidades sobre la comunicación o los datos transportados.

Arquitectura de red: Conjunto de capas y protocolos asociados. Por ejemplo, OSI no es una arquitectura de red porque se definen capas pero no protocolos. TCP/IP sí es arquitectura de red. La condición para que se dé un intercambio de información transparente y eficaz es tener la misma arquitectura de red (tanto en emisor como en receptor).

Pila de protocolos: Especificación en capas de los protocolos que constituyen una arquitectura de red.

Interfaz: Es el medio a través del cual se realiza la comunicación vertical entre dos capas adyacentes.

Puntos de acceso al servicio (SAP): (Service Access Point) puntos sobre los que se realiza la comunicación vertical entre dos capas adyacentes.

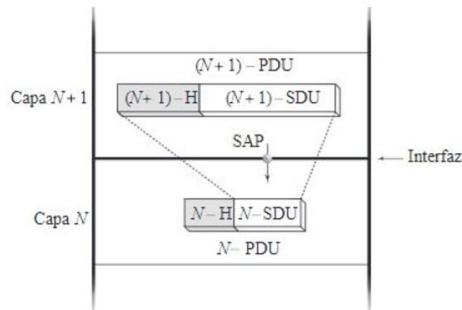
Unidad de datos de servicio (SDU): (Service Data Unit) datos manejados por la entidad y que proceden de la capa inmediatamente superior.

Unidad de datos de protocolo (PDU): (Protocol Data Unit) relativa a la SDU recibida de la capa superior más la cabecera añadida, a efectos de llevar a cabo la función específica desarrollada en colaboración con la entidad o entidades paritarias. Existen:

- **TPDU:** PDU de transporte
- **SSDU:** SDU de sesión
- **NSDU:** SDU de red

De este modo un TSAP no es más que un SAP situado en la interfaz de sesión-transporte, conocidos como sockets.

Un NSAP es uno situado en la interfaz transporte-red



Comunicación vertical. Interfaz y SAP.

Retardos en la Comunicación:

- Procesamiento nodal
- En cola: esperando la transmisión
- Transmisión: transmitir tramas a una velocidad de transmisión
- Propagación: metros que se recorren, relacionada también con la velocidad de propagación

Ttransmisión : $T_t = L(\text{bits}) / V_t (\text{bps}) \rightarrow V_t = C = \text{BW}$

Tpropagación : $T_p = D(m) / V_p (\text{m/s})$

Servicios:

Los servicios ofrecidos por cada una de las capas pueden ser de dos tipos:

- **Orientado a conexión (SOC):** Se establece la conexión como paso previo a la transmisión de datos entre el emisor y el receptor. Por ejemplo, una llamada de teléfono (se tiene que establecer la conexión antes de poder hablar).
- **No están orientadas a conexión (SNOC):** No requiere conexión previa. Por ejemplo, enviar una carta.

Un servicio puede ser confirmado(fiable) si el emisor tiene constancia de la recepción del destino o no confirmado(no fiable), en cuyo caso no se produce la realimentación necesaria acerca de este hecho. Ejemplo: un envío postal normal (servicio no confirmado) frente a uno certificado (servicio confirmado).

Primitiva de servicio: Cada uno de los procesos elementales en que se desarrolla un servicio. Hay 4, todas se emplean en el desarrollo de un servicio orientado a conexión pero solo las dos primeras se usan para un servicio no orientado a conexión.

Las cuatro en total se usan en un servicio orientado a conexión.

- **Solicitud (Request):** Requerimiento de servicio por parte de un solicitante. Por ejemplo, al marcar un número de teléfono.
- **Indicación (Indication):** Recepción de la solicitud en el destino. El ring del teléfono.
- **Respuesta (Response):** Si el destino acepta el servicio, responde a la solicitud. El receptor descuelga el teléfono.
- **Confirmación (Confirm):** El emisor se percata de que el teléfono ha sido descolgado en el otro extremo.

1.4 Internet: Topología y Direccionamiento

Intranets (Ethernet-WIFI) del usuario: zona pública y privada.

- Zona pública: IP pública, fijas y únicas
- Zona privada: IP privada, se repiten

Redes de acceso: infraestructura para dar servicio (proveedor de acceso FIBRA/ADSL) y pertenece a un operador (vodafone, telefónica...)

Redes troncales: grandes operadores de telecomunicaciones

Entre operadores existen dos acuerdos:

- **Peering** → no hay contraprestación económica, acuerdo de pares (tú me das yo te doy). El: grandes operadores (muy grandes) como telefónica
- **Tránsito** → hay contraprestación económica (yo te dejo tú me pagas)

Tres categorías de operadores:

- **Tier1:** capaces de alcanzar cualquier IP del mundo con acuerdos de peering. MUY GRANDES. No pagan a nadie (ej: telefónica)
- **Tier2:** tienen ciertos acuerdos de peering y también de tránsito. GRANDES
- **Tier3:** solo contratos de tránsito. Tienen que pagar para acceder a la IP del mundo.

Los Tier1 les venden tránsito a los Tier2.

- Puntos neutros ó PoP (Point of Presence) ó IXP (Internet eXchange Point): intercambiar tránsito entre operadores (sin ánimo de lucro)

Direccionamiento: metodología para identificar entidades. Tipos de direccionamiento:

- **URL:** en la capa de aplicación.
- **Puertos:** identificar el proceso origen y el destino en la capa de transporte.
- **Dirección IP:** identifica los hosts en la capa de red.

Descubre la gama ZBook para creativos.

En Septiembre con 6% dto. adicional sólo para ti. Usa el cupón: **WUOLAH6**

Últimos días del cupón



Tema 2

Servicios Y Protocolos De Aplicación En Internet

2.1 Introducción a las Aplicaciones de Red

Modelo Cliente-Servidor

Un **servidor** es un proceso que permite el acceso remoto a ciertos recursos existentes en un host. Una Aplicación servidora se caracteriza por encontrarse inicialmente en un estado latente de escucha, no realizan función alguna hasta que recibe una solicitud remota por parte del proceso remoto o cliente.

En la capa de transporte, concretamente en el caso de TCP, los clientes son los que hacen la apertura activa, mientras que los servidores están permanentemente monitoreando el correspondiente puerto a la espera de recibir las solicitudes generadas por los clientes.

En la capa de aplicación, cliente es el proceso a través del cual el usuario interacciona con el servidor que proporciona el servicio concreto ofertado. El modelo cliente-servidor está muy asociado a interacciones solicitud-respuesta. Así el intercambio de información se realiza mediante mensajes originados en el cliente y enviados al servidor, es decir, las solicitudes ante las cuales los servidores devuelven un mensaje, es decir la respuesta.

Un proceso Cliente-Servidor sigue los siguientes pasos:

1. Se inicia la aplicación servidora que pasa a modo escucha (puesta en marcha pasiva)
2. La ejecución de la aplicación cliente dará lugar al envío de una solicitud de servicio hacia el servidor. Esto supone una puesta en marcha activa del servicio.
3. El servidor responderá a dicha solicitud de acuerdo con la aplicación desarrollada, llevándose a cabo el intercambio de información.
4. Tras el desarrollo del servicio el servidor volverá al estado de escucha y el cliente finalizará su ejecución.

Otra característica diferenciadora es que mientras que al servidor se le suele asociar un puerto conocido a priori, los clientes usan puertos diferentes dependiendo de las disponibilidades del SO en cuestión.

Las aplicaciones cliente servidor pueden implementarse en forma de cascada o recursiva, una entidad servidor puede actuar a su vez como cliente de otra servidora. También pueden implementarse en hosts distintos o en el mismo, por lo general ambas se ejecutan en el plano de usuario. Al contrario los servicios de la capa de transporte e inferiores de la pila TCP/IP son parte integrante del SO.

Las aplicaciones (cliente o servidora) hacen uso de los servicios de TCP/IP mediante un conjunto de llamadas al sistema, son un conjunto de procedimientos que ceden el control al SO a través de las cuales las aplicaciones pueden comunicarse entre sí , este conjunto de llamadas al sistema define la interfaz entre la capa de aplicación y la pila TCP/IP.

Vamos a destacar dos interfaces Winsock para entornos Windows y Socket para sistemas Unix.

No podemos confundir interfaz con protocolo, la interfaz está relacionada con el SO e incluso con el compilador, mientras que el protocolo es el conjunto de reglas sintácticas y funcionales que regulan el intercambio de información entre entidades pares.

Interfaz Socket

Para interaccionar con una aplicación remota a través de la interfaz socket, la aplicación debe abrir la comunicación invocando la correspondiente llamada al sistema, la cual devuelve un descriptor (denominado **socket**), luego se escribe en y se lee de él y por último se cierra. Un socket es una “**puerta**” entre la aplicación y los servicios de **transporte**. El descriptor de comunicación o socket consiste en realidad en una **variable de tipo puntero a una estructura**, la cual tiene definida una serie de campos que caracterizan o definen todas las propiedades y atributos necesarios asociados a la comunicación. Entre otras, esta estructura tiene definidos los siguientes campos:

- **Familia de protocolos:** referente al tipo de socket que trata. (PF_INET para IPv4)
- **Servicio:** Tipo de servicio a utilizar ofrecido por las capas inferiores (SOCK_STREAM indica TCP...)
- **IP local:** Dirección IP del host local.
- **IP remota:** Dirección IP del host remoto.
- **Puntero local:** número de puerto asociado a la entidad local.
- **Puerto remoto:** número de puerto asociado a la entidad remota.

Cuando se crea un socket con el comando open el SO le asocia internamente el primer descriptor libre, estando inicialmente vacíos los datos de la estructura apuntada. Para manejar las siguientes direcciones se utiliza:

- **Direcciones IPv4:** estructuras sockaddr_in, la familia de direcciones es AF_INET (valor 2)
- **Direcciones IPv6:** estructura sockaddr_in6, la familia de direcciones es AF_INET6 (valor 10)

Retardo en Cola

Para estimar los retardos en cola se usa la teoría de colas: El uso de un servidor se modela un sistema M/M/1.

- M -> cómo es la distribución de los tiempos de llegada
- M -> distribución de probabilidad del tiempo de servicio
- 1 -> servidores

El retardo en cola es: Donde T_s (distribución exponencial) es el tiempo de servicio y λ (Poisson) el ratio de llegada de solicitudes. Esta expresión se puede utilizar para calcular el retardo en cola en un router.

$$R = \frac{\lambda(T_s)^2}{1 - \lambda \cdot T_s}$$

¿Qué Definen los Protocolos de Aplicación?

- **Tipo de servicio:** Orientado o no a conexión, Realimentado o no ...
- **Tipo de mensaje:** Request, response...
- **Sintaxis:** Definición y estructura de campos en el mensaje. En aplicación generalmente son orientados a texto (HTTP), aunque hay excepciones(DNS). Tendencia a usar el formato Type-Length-Value.
- **Semántica:** Significado de los campos.
- **Reglas:** Cuándo los procesos envían mensajes/responden mensajes.

Tipos de protocolos:

Protocolos de dominio público: abiertos, sin ánimo de lucro. Ej: HTTP,SMTP...

Protocolos propietarios/privados: ej: Skype,IGRP...

Protocolos in-band: mismo socket para enviar la información de señalización (info de control) y datos (contenido del fichero). Ej: HTTP

Protocolos out-of-band: utilizan una conexión para enviar la información de señalización (info de control) y otra para los datos (contenido del fichero). Ej: FTP

Protocolos persistentes: se envían todas las solicitudes y sus correspondientes respuestas a través de la misma conexión TCP. Mantienen la conexión al servir a más de un objeto.

Protocolos no persistentes: cada par solicitud/respuesta se envía a través de una conexión TCP separada

Las solicitudes pueden ser:

- State-less → ligeros en cuanto a recursos en el servidor, sin guardar el historial. Ej: HTTP
- State-full → guardan el historial. Ej: Correo

La tendencia es hacer protocolos flexibles con:

- Una cabecera fija.
- Una serie de "trozos" (obligatorios y opcionales) Estos trozos pueden incluir una cabecera específica más una serie de datos en forma de parámetros (Parámetros fijos, parámetros de longitud variable u opcionales). Estos parámetros usan el formato TLV Type-Length-Value.

Aplicaciones de Red. Características

- **Tolerancia a pérdidas de datos (errores):** Algunas apps pueden tolerar algunas pérdidas de datos; otras (FTP,Telnet...) requieren transferencia 100% fiable.

- **Existencia de requisitos temporales:** Algunas apps denominadas inelásticas requieren retardo (delay) acotado para ser efectivas, otras no
- **Demanda de ancho de banda (tasa de transmisión o throughput):** Algunas apps requieren envío de datos a una tasa determinada, otras no.
- **Nivel de seguridad:** Los requisitos de seguridad para las distintas apps son muy variables (encriptación, autenticación ...)

Conclusión: Las distintas aplicaciones tienen requisitos HETEROGÉNEOS.

Como las demandas son muy heterogéneas y no todas necesitan los mismos requisitos, se usa el servicio TCP(Protocolo de Control de Transmisión) y UDP (User Datagram Protocol). Ambos son servidores multiprotocolo (permiten que el servicio proporcionado pueda ser tanto orientado a conexión como no orientado a conexión). ¿En qué se diferencian?

Servicio TCP

El modelo de servicio TCP incluye un servicio orientado a la conexión y al transporte fiable de datos con control de errores. Cuando una aplicación utiliza TCP como protocolo de transporte, recibe estos dos servicios de TCP:

- **Servicio orientado a conexión:** TCP tiene al cliente y al servidor intercambian información de control sobre la capa de transporte, del uno al otro antes de que los mensajes de la capa de aplicación empiecen a fluir. A esto se le llama handshaking y avisa al cliente y al servidor, preparándolos para el intercambio de paquetes. Después de esta fase, se dice que existe una conexión TCP entre los sockets de dos procesos. Es una conexión full-duplex, ya que los procesos pueden enviarse mensajes el uno al otro al mismo tiempo. Cuando la aplicación termina el envío de mensajes, se debe cerrar la conexión.
- **Transferencia de datos fiable:** El proceso de comunicación puede depender de que TCP haga el envío de todos los datos sin errores y en el orden correcto. Cuando un lado de la aplicación pasa un flujo de bytes a un socket, puede contar con que TCP enviará el mismo flujo de bytes al socket receptor sin pérdidas de bytes ni bytes duplicados.

TCP incluye mecanismos de control de congestión y control de flujo.

El **mecanismo de control de congestión** consiste en acelerar el envío de paquetes hasta que se sature la red. Si la red se satura comenzará a descartar paquetes, que tendrán que ser retransmitidos, lo cual puede incrementar aún más la saturación de la red, y cuando llegue al punto en el que la red se encuentre saturada, se reducirá la tasa de envío para reducir la saturación (este algoritmo se llama Slow-start). Ofrece la mayor velocidad posible dependiendo de las máquinas que haya en la red para evitar que haya unas pocas que dejen sin red a las demás.

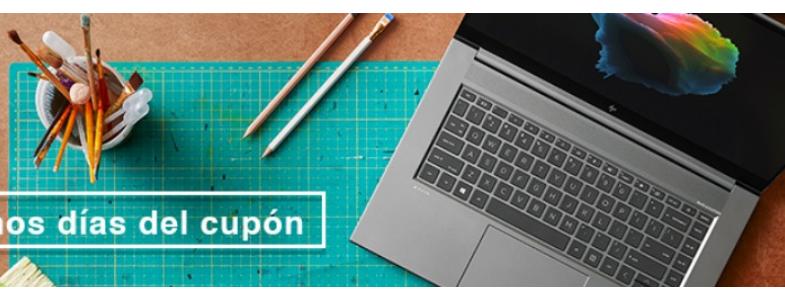
Servicio UDP

UDP es un protocolo de transporte ligero que provee de servicios mínimos, es no orientado a conexión así que no hay handshaking antes de que comience el proceso de

Descubre la gama ZBook para creativos.

En Septiembre con 6% dto. adicional sólo para ti. Usa el cupón: **WUOLAH6**

Últimos días del cupón



comunicación. UDP proporciona una transferencia no segura de datos, esto significa que cuando un proceso envía un mensaje a un socket UDP, UDP no garantiza que el mensaje llegue al proceso receptor. Además los mensajes que llegan al receptor pueden llegar desordenados.

UDP no incluye mecanismos de control de congestión, ni mecanismos de control de flujo.

Al ser TCP y UDP (capa de transporte) usuarios del protocolo IP (capa de red) no garantizan Calidad de Servicio QoS, es decir:

- El retardo no está acotado.
- Las fluctuaciones en el retardo no están acotadas.
- No hay una velocidad de transmisión mínima garantizada.
- No hay probabilidad de pérdidas acotada

Tampoco hay garantías de seguridad. Se puede mejorar la seguridad de TCP implantando SSL (Secure Sockets Layer) que es un mecanismo que proporciona comunicaciones seguras por una red utilizando protocolos criptográficos.

2.2 Servicio de Nombres de Dominio (DNS)

A pesar de la notación decimal con puntos adoptada para las direcciones IP, es claro que el manejo directo de este tipo de direccionamiento, por parte del usuario es incómodo. Resulta mucho más fácil el manejo de nombres asociados a dichas direcciones. Aceptada esta solución, es preciso arbitrar algún método que posibilite la conversión automática de nombres a direcciones IP, servicio que se conoce como resolución de nombres. Con este objetivo principal surgió el denominado sistema de nombres de dominio (DNS). Ej: convierten www.ugr.es a 150.214.204.25.

Sus características principales son las siguientes:

- Se suele invocar como paso previo a la ejecución de otros servicios, puesto que la activación de cualquiera de ellos por el cliente correspondiente utiliza usualmente el nombre del servidor con el que se desea contactar. Desde este punto de vista antes de llevar a cabo el servicio solicitado como tal es necesario obtener la dirección IP del servidor a partir de su nombre, o lo que es lo mismo, resolver su nombre. Por supuesto, todo servicio puede desarrollarse a través de su dirección IP.
- Es transparente al usuario, es decir, el proceso de resolución se lleva a cabo de forma automática por la aplicación cliente ejecutada, sin necesidad de que lo solicite explícitamente.
- El espacio de nombres de dominio tiene una estructura jerárquica en forma de árbol centrada en el dominio raíz (root) ".": Parte_local.dominio_niveln.dominio_nivel2.dominio_nivel1
- A los dominios de nivel1 se les denomina Top Level Domains (TLD) (.com, .edu, .es etc...)



WUOLAH

- El dominio raíz (“.”) está gestionado por el ICANN (Internet Corporation for Assigned Names and Numbers). ICANN delega la gestión de algunos dominios TLD a centros regionales.

El servicio DNS se implementa tanto sobre UDP como TCP, es un ejemplo de servidor multiprotocolo.

Al igual que el espacio de nombres de dominio, el servicio de resolución es jerárquico, tal que los distintos servidores se organizan en un árbol lógico con una estructura coincidente con la jerarquía o árbol formado por el espacio de nombres de dominio. Así, un host contactará con el servidor DNS establecido al efecto, el cual a su vez podrá contactar (si no conoce la respuesta a la consulta recibida) con un servidor DNS.

- De nivel superior, si el dominio sobre el que se realiza no depende jerárquicamente de él.
- De nivel inferior, en caso contrario.

DNS: Protocolo de aplicación para el acceso a una base de datos distribuida con una gestión distribuida. 3 niveles de servidores, asociados al árbol de nombres de dominio:

- Servidores raíz “.” (Root servers)
- Top-Level Domain (TLD servers) -> servidores de 1 nivel (.com,.edu...)
- Servidores Locales (Local servers) -> .ugr,.tstc (nivel más bajo)

Buscamos en google www.google.com:

1. Le pasa el control a “resolver” (proceso dentro de la máquina que resuelve la IP), tiene una caché local. Si no está el registro buscado en la caché hace una consulta al servidor (DNS local server). Gethostbyname() es una llamada a una rutina que pasa el control al SO, se le pasa el nombre de dominio (www.ugr.es) y hasta que no devuelve la IP no se le devuelve el control. Una IP puede tener muchos nombres asociados.
2. ¿Cómo sabe el server su IP? Mediante el DHCP o manualmente en la configuración
3. Hace la consulta al root server, puede saber la respuesta o no. En el caso de que no haga un análisis sintáctico.
4. Baja tantos niveles como sea necesario para encontrar la respuesta

Desde este punto de vista la resolución de nombres puede ser recursiva o no recursiva (interactiva). En el primer caso se deja la interacción abierta durante todo el procedimiento. En cambio la resolución no recursiva se caracteriza porque cuando un servidor no conoce la respuesta a la resolución planteada, este responde proporcionando la identidad del servidor siguiente en la jerarquía, con el que primero debe contactar directamente para hacer la consulta pertinente.

En este caso, la interacción no se mantiene durante todo el procedimiento, sino que cesa una vez proporcionada la respuesta. La resolución recursiva consume más recursos, si bien tiene como ventaja que la respuesta puede ir actualizando las memorias caché de todos los

servidores involucrados; por el contrario la resolución iterativa consume menos recursos, si bien las caché no pueden ser actualizadas.

Conceptos y Gestión de la Base de Datos:

El sistema DNS está formado por un conjunto de servidores cooperativos, organizados en árbol, que almacenan parcialmente la BD, denominado BIND (Berkeley Internet Name Domain). Cada servidor es responsable de una zona

- **Zona:** conjunto completo de nombres de dominio que están por debajo de un nodo del árbol. En cada zona existe un servidor DNS responsable que tiene la obligación de conocer la resolución de todos los posibles nombres que pertenezcan a esa zona.
- Este servidor se conoce como la **autoridad** y las respuestas proporcionadas por él se denominan **autorizadas**.
- En cada zona hay servidores primarios (almacenan una copia master de la db en discos locales) y servidores secundarios (obtienen la db por transferencia).
- Para ganar escalabilidad se define el concepto de **delegación de autoridad**, existen 13 servidores raíz (de A a M) con múltiples instancias (réplicas con la misma IP) repartidas en el mundo.
- El root-server F (y otros) tiene un servidor en Madrid (Espanix:punto neutro)
- Además existe un servicio de caché para mejorar las prestaciones
- Una autoridad para una zona dada puede liberarse de la obligación de conocer y gestionar parte de su zona **cediendo esta obligación** a un **servidor delegado** que se responsabilizará de esta obligación una vez que la autoridad haya sido delegada.

Por tanto las resoluciones cuya respuestas no sean conocidas por el primer servidor de nombres son redirigidas al raíz. Este tras realizar un análisis sintáctico del nombre del dominio en cuestión, obtiene el nombre del primer nivel. A partir de este momento si tienen delegada la autoridad para esa zona y no tiene en cache la resolución trasladará hacia abajo en el árbol de resolución hacia la autoridad responsable de la zona delegada. Este procedimiento se repite hacia abajo tantos niveles como hagan falta hasta que, o bien se encuentre la solución proveniente de la memoria caché de un servidor, o bien se llega a la autoridad responsable de la zona correspondiente, la cual, si no ha delegado en sucesivos niveles de autoridad, tiene la obligación de conocer la respuesta asociada a la resolución de nombres planteada.

Podemos clasificar las respuestas así:

- **Respuesta con autoridad:** El servidor tiene autoridad sobre la zona en la que se encuentra el nombre solicitado y devuelve la IP
- **Respuesta sin autoridad:** El servidor no tiene autoridad sobre la zona en la que se encuentra el nombre solicitado, pero lo tiene en caché.
- **No conoce la respuesta:** El servidor preguntará a otros servidores. Normalmente se "eleva" la petición a uno de los servidores raíz.

Es por esto que se dice que DNS es un protocolo de aplicación para el acceso a una base de datos distribuida con una gestión distribuida.

¿Cómo es la BD de DNS?

- Se almacena en ficheros de texto (txt) denominados zone files.
- Cada zone file contiene registros denominados Resource Record
- Cada zone file define un TTL (Time to Live): tiempo de validez de los RRs en cache
- Cada RR contiene:
 - Nombre del dominio: nombre del dominio al que se refiere el RR.
 - Clase: en Internet siempre IN.
 - Tipo: Tipo de registro.
 - + **SOA** Registro (Start Of Authority) con la autoridad de la zona.
 - + **NS** Registro que contiene un servidor de nombres.
 - + **A** Registro que define una dirección IPv4.
 - + **MX** Registro que define un servidor de correo electrónico.
 - + **CNAME** Registro que define el nombre canónico de un nombre de dominio.
 - + **HINFO** Información del tipo de máquina y sistema operativo
 - + **TXT** Información del dominio.
 - + **PTR** Registro que contiene un nombre de dominio (para resoluciones inversas) o Valor: Contenido que depende del campo tipo

Existe una BD asociada de resolución inversa para traducir direcciones IP en nombres de dominio. Cuando un cliente DNS quiere obtener el nombre de dominio correspondiente a una IP, hace una resolución inversa, preguntando el RR tipo PTR asociado a la dirección.

Los mensajes que usa este protocolo son de 2 tipos, y las peticiones y las respuestas tienen el mismo formato.

- **Cabecera:** 12 bytes que almacenan información como la cantidad de información que aparecerá en los siguientes campos.
- **Identificador:** Cadena de 16 bits que identifica al mensaje
- **Sección de petición:** contendrá información sobre una o más peticiones que se van a realizar, tendrá un nombre(será del que se solicite la información) y tipo (información que se pida).
- **Sección de respuesta,** donde irá la respuesta a una petición
- **Sección de autoridad** (contendrá registros de servidores autoridad)
- **Sección adicional** (otra información).

2.3 La Navegación Web

El HyperText Transfer Protocol (HTTP) está implementado en dos programas, el cliente (buscador web) y el servidor instalados en distintos equipos, ambos programas se comunican mediante el intercambio de mensajes HTTP. El protocolo define la estructura de estos mensajes y de cómo el cliente y el servidor deben intercambiarlos.

Una página Web es un documento formado por objetos: ficheros HTML, imágenes, ficheros de audio o video ... todo esto accesible mediante una única dirección URL (El fichero HTML referencia al resto de ficheros mediante sus respectivas URLs).

Descubre la gama ZBook para creativos.

En Septiembre con 6% dto. adicional sólo para ti. Usa el cupón: **WUOLAH6**

Últimos días del cupón



Las páginas webs pueden ser:

- Estáticas (contenido invariable)
- Dinámicas (contenido variable), pueden proporcionar contenido variable:
 - Usando lenguajes de scripting de cliente: JavaScript por ejemplo
 - Usando lenguajes de scripting en el servidor: Perl, PHP, Ruby... Se utilizan insertando etiquetas dentro de la página web. Cuando el cliente solicita la página web, el servidor web interpreta estas etiquetas para realizar acciones en el servidor generando contenido dinámico.

Características del protocolo HTTP:

- Usa los servicios de TCP (S.O.C) en el puerto 80 -> inicio de una conexión TCP, envío de mensajes HTTP y cierre de conexión TCP.
- HTTP es “stateless” (sin estado) -> el servidor no mantiene información sobre las peticiones de los clientes (su estado) y así ahorra recursos, aunque hace más compleja la interacción. Usa Cookies
- Protocolo in-band orientado a texto

Podemos diferenciar entre conexiones persistentes (se realiza todo bajo la misma conexión TCP) y conexiones no persistentes (cada par petición respuesta se realiza bajo una conexión TCP distinta). Por defecto HTTP utiliza conexiones persistentes, pero tanto el cliente como el servidor se pueden configurar para utilizar conexiones no persistentes.

Conexiones no Persistentes

Si se va a acceder a una página web que consiste en 1 HTML y 10 JPEGs, se tendrán que realizar 11 conexiones TCP, la ventaja de este tipo de conexiones es que no tienen porqué realizarse en serie, hay navegadores hoy en día que soportan 5 o hasta 10 conexiones TCP en paralelo.

El Round-trip time(RTT) es el tiempo en el que un paquete viaja del cliente al servidor y vuelve. Incluye la propagación del paquete, tiempo de encolamiento y tiempo de procesamiento. Normalmente se tarda 2 veces el RTT más el tiempo de transferencia del archivo.

Conexiones Persistentes

Las conexiones no persistentes tienen algunos inconvenientes, primeramente se debe establecer una conexión nueva por cada petición de un objeto, para cada una de estas conexiones se tienen que alojar los buffers TCP, se deben crear las variables tanto en el cliente como en el servidor (Esto puede suponer bastante para un servidor que reciba cientos de peticiones de diferentes clientes simultáneamente). Además como hemos dicho antes, cada petición sufre un retraso de dos RTT, uno para establecer la conexión TCP y otro para la petición y la recepción del objeto.

Con las conexiones persistentes todo el paso de peticiones/respuestas se hace mediante la misma conexión TCP. Incluso se podrían enviar distintas páginas web que contenga el servidor por el mismo canal TCP. También se podrían hacer distintas peticiones sin haber

recibido aún la respuesta a otras peticiones(pipelining). Se cerrará la conexión TCP cuando pase un tiempo sin envío de peticiones HTTP (este tiempo es configurable).

Formato de Mensajes HTTP

1. El cliente HTTP (navegador) solicita un objeto identificado por su URL, en el ejemplo www.ugr.es/pages/Universidad. Según la configuración del servidor, si no se especifica nada, por defecto se sirve el fichero index.html.
 - a. El Cliente HTTP inicia conexión TCP al servidor HTTP (proceso) en www.ugr.es en el puerto 80 (segmento SYNC de TCP sin datos)
 - b. El Servidor HTTP acepta la conexión y solicita al cliente abrir la conexión (SYNC+ACK)
 - c. El cliente confirma (ACK)
2. El cliente consulta al resolver de DNS por la dirección IP de www.ugr.es
3. DNS contesta 150.214.204.231
4. El cliente abre una conexión TCP al puerto 80 de 150.214.204.231 (3 bandas)
5. El cliente envía una petición “GET /pages/universidad/ ...” (más otra información adicional: cabeceras, cookies, variables, etc)
6. El servidor responde enviando el fichero “index.html” por la misma conexión TCP
7. Al usar TCP el cliente y servidor de HTTP reciben un servicio orientado a conexión, fiable, sin errores, con control de flujo, con control de congestión, etc. Es decir una comunicación TRANSPARENTE y FIABLE.
8. Si es persistente se siguen solicitando objetos de la página (“GET...”) por la conexión
9. Se cierra la conexión TCP y se liberan recursos en el servidor y cliente
10. El cliente visualiza el contenido

Hay dos tipos de mensajes HTTP, las peticiones y las respuestas:

- **Request(peticiones):** La primera línea de los mensajes de petición se llama request line, esta línea tiene 3 campos, el método(GET, POST, HEAD, PUT, DELETE), la URL y la versión de HTTP que se está usando. El método más usado es GET, en este método se le hace una petición al servidor pidiéndole el objeto identificado por la URL.

MÉTODOS (acciones solicitadas por los clientes en los request messages):

- **OPTIONS:** solicitud de información sobre las opciones disponibles
- **GET:** solicitud de un recurso (puede ser condicional)
- **HEAD:** igual que GET pero el servidor no devuelve el “cuerpo” sólo cabeceras
- **POST:** solicitud al servidor para que acepte y subordine a la URI especificada, los datos incluidos en la solicitud
- **PUT:** solicitud de sustituir la URI especificada con los datos incluidos en la solicitud.
- **DELETE:** solicitud de borrar la URI especificada.

CABECERAS (47 request headers y 49 response headers)

Las líneas siguientes a la línea de petición (request line) se llaman **headers lines** (**líneas de cabecera**).

- **Host**, especifica el host donde reside el objeto (esto es necesario para la caché del proxy).
 - **Connection: close**, esta línea le dice al servidor que no es necesaria una conexión persistente y que por tanto cierre la conexión tras el envío del objeto requerido.
 - **User-agent** especifica el navegador utilizado.
 - **Accept-language** sirve para que si hay varias versiones de un objeto se mande la que corresponda con el lenguaje requerido.
 - **Entity body** que es usado solo en las peticiones POST, donde el servidor manda cierta información conforme lo que el usuario haya introducido (por ejemplo un formulario).
 - **Content-type**: descripción MIME de la información contenida en este mensaje
 - **Content-length**: longitud en bytes de los datos enviados, expresado en base decimal
 - **Content-encoding**: formato de codificación de los datos enviados en este mensaje. Sirve, por ejemplo, para enviar datos comprimidos y encriptados
 - **Date**: fecha local de la operación. Las fechas deben incluir la zona horaria en que reside el sistema que genera la operación. No existe formato único de fechas
-
- **Response(respuestas):** Este mensaje tiene 3 secciones, la línea de estado (**status line**), seis líneas de cabecera(**header lines**) y el **entity body**(en esta última va el significado del mensaje, el objeto requerido). La línea de estado tiene tres campos, la versión del protocolo, un código de estado y el mensaje de estado correspondiente. En las líneas de cabecera tenemos un campo **Connection:close** (igual que en peticiones), otro **Date** que indica cuando se creó la respuesta. **Server** indica que Apache Web Service generó la respuesta (equivalente al User-agent). **Last-Modified** indica cuando el objeto fue creado o modificado por última vez. **Content-Length** (número de bytes). **Content-Type** indica el tipo de contenido en el entity body.

CÓDIGOS DE RESPUESTA (para los response messages del servidor):

- 1xx indican mensajes exclusivamente informativos
- 2xx indican algún tipo de éxito
- 3xx redirección al cliente a otra URL
- 4xx indican un error
- 5xx indican un error

Cookies

Las cookies son pequeños ficheros de texto que se intercambian los clientes y servidores HTTP, para solucionar una de las principales deficiencias del protocolo, la falta de información de estado entre dos transacciones. La primera vez que un usuario accede a un determinado documento de un servidor, éste proporciona una cookie que contiene datos que relacionarán posteriores operaciones. El cliente almacena la cookie en su sistema para usarla después. En los futuros accesos a este servidor el navegador podrá proporcionar la cookie original, que servirá de nexo entre este acceso y los anteriores. Todo este proceso es automático sin intervención del usuario.

Una cookie es simplemente una serie de líneas de texto, con pares variable/valor. Existe un conjunto predefinido de nombres de variable, necesarias para el correcto funcionamiento de las cookies, estas son el **dominio** (conjunto de direcciones web para el que es válida), **path** (fija el subconjunto de URLs para las que sirve), **versión** (versión del modelo de cookies), **expires** (fecha de expiración de los datos).

Un servidor envía los diferentes campos de una cookie con la cabecera HTTP Set-Cookie.

Proxy Server (Web Cache)

Un servidor proxy es un servidor intermedio entre el cliente y el servidor que contiene respuestas a peticiones HTTP realizadas recientemente.

Primero el navegador establece una conexión TCP con el servidor proxy y manda una petición para el objeto requerido, el servidor proxy comprueba si existe una copia del objeto alojada en el servidor, si la encuentra le manda una respuesta al buscador del cliente con la copia, si no la encuentra el proxy establece una conexión TCP con el servidor principal y le manda una petición del objeto, seguidamente el servidor le manda el objeto al proxy(donde se almacena el objeto), y el proxy finalmente a través de una respuesta HTTP le manda una copia del objeto al buscador.

Estos servidores proxy son normalmente instalados por los proveedores de red. Esto ha sido desarrollado por dos razones, puede reducir el tiempo de respuesta ya que el ancho de banda entre el cliente y el proxy es mayor que el ancho de banda entre el cliente y el servidor, y además puede reducir significativamente el tráfico del servidor principal eliminando las peticiones repetidas.

Get condicional

Es una petición que realiza el servidor proxy al servidor principal para comprobar que la copia que tiene en memoria se trata de una copia válida, es una simple petición GET acompañada de un campo If-Modified-Since con una fecha, si este campo coincide con el Last-Modified, el servidor mandará una respuesta con el mensaje 304 Not Modified, si no coincide el servidor principal mandará el nuevo dato.

Descubre la gama ZBook para creativos.

En Septiembre con 6% dto. adicional sólo para ti. Usa el cupón: WUOLAH6

Últimos días del cupón



2.4 El Correo Electrónico

Podemos identificar dos entidades:

- **Agente de usuario(MUA):** Interfaz entre el sistema de correo y el usuario, permitiendo este último la lectura y escritura de correos electrónicos. También se llaman clientes de mensajería.
- **Agente de transporte(MTA):** Encargado de transmitir el correo electrónico interno y externo al sistema, este módulo también se conoce como estafeta de correo.

Un destinatario de correo se identifica mediante una dirección de correo electrónico, generalmente de la forma nombre@dominio.

En el servicio de correo se identifican dos acciones diferenciadas que involucran a dos o más protocolos. Por un lado se encuentra el **procedimiento de envío de correo** que puede ser ejecutado tanto el MUA como en el MTA. En este caso el protocolo especificado es **SMTP(Simple Mail Transfer Protocol)** y el **procedimiento de consulta o lectura del buzón de correo entrante**. En este caso hay dos protocolos especificados, **POP e IMAP** que pueden usarse de forma indistinta. Además de los anteriores, **WebMail** es una alternativa que simplifica la administración ya que reduce el número de puertos abiertos necesarios, consiste en usar HTTP como contenedor de los mensajes correspondientes a POP, IMAP o SMTP.

Formato de los mensajes email

Todo email consta de dos partes:

- **Cuerpo:** datos del mensaje.
- **Datos administrativos:** Conjunto de información necesaria para el envío y la gestión del mensaje.
 - + De transporte: destinatario y remitente
 - + Otros datos como la fecha

Protocolo Simple de Transferencia de Correo Electrónico (SMTP)

Sus principales características son:

- Inicialmente RFC 821, luego RFC 2821 y 5381 seguido de actualizaciones RFC 5335 y 5336
- Utiliza TCP, además su puerto estándar reservado es el 25.
- A pesar de la consideración ocasional de pasarelas de correo (mail gateway), dispositivos intermediarios a los que se recurre ocasionalmente para el almacenamiento temporal y retransmisión de los mensajes, la transmisión de los mensajes email se suele realizar mediante una conexión directa entre los MTA origen y destino involucrados.
- SMTP es un protocolo orientado a conexión, es in-band y es state-full: implica tres fases
 - Handshaking ("saludo")
 - Transferencia de mensajes
 - Cierre
- La interacción entre cliente SMTP y servidor SMTP se realiza mediante comandos (texto ASCII) y respuesta (código de estado y frases explicativas)

- Inicialmente los mensajes estaban codificados en ASCII de 7 bits!! -> Con la definición posterior de las extensiones MIME se puede enviar ASCII de 8 bits y formatos enriquecidos

Como otros protocolos de aplicación SMTP está orientado a texto. Es decir, el cliente formula solicitudes o comandos hacia el servidor, una vez establecida la conexión TCP correspondiente entre ambos; y el servidor devuelve al cliente el resultado de la ejecución de la solicitud previa.

Pasos en el envío/recepción de correo:

1. El usuario origen compone mediante su Agente de Usuario (MUA) un mensaje dirigido a la dirección de correo destino.
2. Se envía con SMTP el mensaje al servidor de correo(MTA) del usuario origen que lo sitúa en la cola de mensajes salientes.
3. El cliente SMTP abre una conexión TCP con el servidor de correo (MTA) del usuario destino.
4. El cliente SMTP envía el mensaje sobre la conexión TCP.
5. El servidor de correo del usuario destino ubica el mensaje en el mailbox del usuario destino.
6. El usuario destino invoca a su Agente de Usuario(MUA) para leer el mensaje utilizando POP3, IMAP o HTTP.

Cabe remarcar que la conexión TCP es persistente, es decir, que cuando se abre una comunicación TCP se pueden mandar más de un correo (todos los que el destinatario en cuestión desee enviar a ese servidor).

Extensiones MIME

Las principales limitaciones del SMTP original son que sólo permite el envío de mensajes ASCII y que la longitud de las líneas de datos tienen un límite máximo de 1000 caracteres, por tanto no sería posible el envío de mensajes multimedia. Para resolver esto, se constituyen los que se conoce como Extensiones Multipropósito de Correo Internet (Multipurpose Internet Mail Extensions) en las que se redefine el formato de los mensajes de correo electrónico de manera que:

1. El cuerpo y cabeceras de los mensajes pueden estar formados por caracteres no ASCII.
2. Se acepta un conjunto extensible de formatos para los mensajes.
3. Los mensajes pueden estar constituidos por varias partes independientes y, posiblemente, con formatos distintos.

Algunas de las cabeceras email adicionales definidas por MIME:

- **MIME-Version:** número de la versión de MIME
- **Content-Description:** cadena de texto que describe el contenido del mensaje
- **Content-id:** descriptor de mensaje análogo a Message-Id en SMTP
- **Content-Type:** formato de los datos contenidos en el mensaje
- **Content-transfer-encoding:** codificación utilizada
-

Content-Type-Encoding

Manera en que está envuelto el cuerpo para su transmisión, ya que podría haber problemas con la mayoría de los caracteres distintos de letras, números y signos de puntuación. 5 tipos de codificación: ASCII 7, ASCII 8, codificación binaria, base64 y entrecomillada-imprimible.7.2

TIPOS

- El **tipo application** es un tipo general para los formatos que requieren procesamiento externo no cubierto por ninguno de los otros tipos.
 - **El subtipo octet-stream:** Siendo una cadena de bytes no interpretada
 - **El subtipo Postscript:** Siendo un documento escrito en PostScript
- El **tipo message** permite que un mensaje esté encapsulado por completo dentro de otro. Esto es útil para reenviar correo electrónico.
 - **El subtipo partial** hace posible dividir un mensaje encapsulado en pedazos y enviarlos por separado. Los parámetros hacen posible ensamblar correctamente todas las partes en el destino.
 - **El subtipo external-body** puede usarse para mensajes muy grandes, se da una dirección de FTP y el agente de usuario del receptor puede obtenerlo a través de la red cuando se requiera.
- El **tipo multipart** permite que un mensaje contenga más de una parte, con el comienzo y el fin de cada parte claramente delimitados.
 - **Subtipo mixed:** Cada parte puede ser diferente
 - **Subtipo alternative:** Cada parte tiene el mismo mensaje pero expresado en un medio de codificación diferente.
 - **Subtipo parallel:** Todas las partes deben “verse” simultáneamente.
 - **Subtipo digest:** Cuando se juntan muchos mensajes en un mensaje compuesto.

Protocolos de Acceso o de Correo de Entrega Final (POP3)

Las funciones de un MTA o agente de transporte de correo electrónico son:

- Para el correo local:
 - + Redireccionar mensajes si fuera preciso.
 - + Añadir el mensaje al spool de correo del destinatario.
 - + Devolver los mensajes que no lleguen a su destino, junto con un mensaje de error.
- Para el correo remoto, el MTA debe, además determinar el camino a seguir por el mensaje.

Concluimos en que incluir un MTA en cada host es impracticable ya que consumiría muchos recursos, además implicaría un fuerte acoplamiento temporal. Para solucionar este problema surgen los servicios de correo de entrega final, a través de los cuales se permite a un usuario contactar con un MTA remota a fin de acceder a sus mensajes de email.

Pasemos a estudiar los protocolos de correo de entrega final más extendidos.

POP(Post Office Protocol)

Inicialmente era una especificación denominada download and delete, es decir, tras la descarga del correo electrónico correspondiente a un usuario el host local lo elimina del servidor. Actualmente nos encontramos en la versión POP3. Una vez establecida la conexión TCP entre el cliente y el servidor correspondientes, los estados por los que pasa una conexión POP3 son:

1. **Autorización:** Identificación del cliente.
2. **Transacción:** Estado en el que a través de solicitudes por parte del cliente, se accede al correo electrónico del usuario especificado en el proceso anterior.
3. **Actualización:** Tras la ejecución del comando QUIT por parte del cliente, el servidor elimina los recursos adquiridos durante el estado anterior y cierra la conexión TCP.

Comandos POP3:

Comando	Descripción
USER identification	Este comando permite la autenticación. Debe estar seguido del nombre de usuario, es decir, una cadena de caracteres que identifique al usuario en el servidor. El comando USER debe preceder al comando PASS.
PASS password	El comando PASS permite especificar la contraseña del usuario cuyo nombre ha sido especificado por un comando USER previo.
STAT	Información acerca de los mensajes del servidor
RETR	Número del mensaje que se va a recoger
DELE	Número del mensaje que se va a eliminar
LIST [msg]	Número del mensaje que se va a mostrar
NOOP	Permite mantener la conexión abierta en caso de inactividad
TOP <messageID> <n>	Comando que muestra <i>n</i> líneas del mensaje, cuyo número se da en el argumento. En el caso de una respuesta positiva del servidor, éste enviará de vuelta los encabezados del mensaje, después una línea en blanco y finalmente las primeras <i>n</i> líneas del mensaje.
UIDL [msg]	Solicitud al servidor para que envíe una línea que contenga información sobre el mensaje que eventualmente se dará en el argumento. Esta línea contiene una cadena de caracteres denominada <i>unique identifier listing</i> (<i>lista de identificadores únicos</i>) que permite identificar de manera única el mensaje en el servidor, independientemente de la sesión. El argumento opcional es un número relacionado con un mensaje existente en el servidor POP, es decir, un mensaje que no se ha borrado.
QUIT	El comando QUIT solicita la salida del servidor POP3. Lleva a la eliminación de todos los mensajes marcados como eliminados y envía el estado de esta acción.

IMAP(Internet Message Access Protocol)

Posibilita la gestión virtual del buzón correspondiente como si esta se realizará directamente desde el host servidor. En este sentido IMAP no implica volcado del buzón al host cliente.

La versión actual es IMAP4. Un servicio IMAP se desarrolla en cuatro estados.

- **No-autenticado(NA):** Se ha establecido conexión pero el usuario no se ha autenticado.
- **Autenticado(A):** El usuario se ha autenticado y debe seleccionar un buzón de correo.
- **Seleccionado(S):** El usuario ha seleccionado adecuadamente un buzón de correo.
- **Desconexión(D):** La sesión finaliza, cerrando el servidor la conexión.

Ventajas IMAP

- Permite una gestión de buzones y mensajes más completa.
- La conexión cliente servidor es online aunque se posibilita que sea offline.

Descubre la gama ZBook para creativos.

En Septiembre con 6% dto. adicional sólo para ti. Usa el cupón: WUOLAH6

Últimos días del cupón



- Permite asociar un estado (borrado, leído...) a cada mensaje.
- Permite hacer lecturas parciales de los mensajes.

2.5 Seguridad y Protocolos Seguros

Una red de comunicaciones es segura cuando se garantizan todos los aspectos -> no hay protocolos ni redes 100% seguros. Aspectos:

- **Confidencialidad/privacidad:** el contenido de la información es comprensible sólo por entidades autorizadas.
- **Autenticación:** las entidades son quienes dicen ser.
- **Control de accesos:** los servicios están accesibles sólo a entidades autorizadas.
- **No repudio o irrenunciabilidad:** el sistema impide la renuncia de la autoría de una determinada acción.
- **Integridad:** el sistema detecta todas las alteraciones (intencionadas o no) de la información.
- **Disponibilidad:** el sistema mantiene las prestaciones de los servicios con independencia de la demanda.

¿En qué nivel/capa se debe situar la seguridad? en TODOS.....el grado de seguridad lo fija el punto más débil

- **Ataque de seguridad:** cualquier acción intencionada o no que menoscaba cualquiera de los aspectos de la seguridad
- Tipos de ataques:
 - **Sniffing** = vulneración a la confidencialidad, escuchas (husmear)
 - **Spoofing (phishing)** = suplantación de la identidad de entidades
 - **Man_in_the_middle** = hombre en medio (interceptación). Secuestran una conexión suplantando a los dos extremos
 - **Distributed Denial_of_Service (DDoS)** = contra la disponibilidad, para reducir prestaciones a ciertos servicios. Denegación de servicio distribuido, ejemplo Flooding (inundación)
 - **Malware** = software que hace un uso fraudulento, ilegal, no autorizado. Troyanos, gusanos, spyware, backdoors, rootkits, ransomware, keyloggers

Mecanismos de Seguridad

- **Cifrado de Datos:** Pretende dotarnos de mecanismo que garanticen la confidencialidad, que la información solo sea comprensible por entidades autorizadas. Algoritmo conocido del cual desconocemos las claves
 - **Cifrado simétrico:** algoritmos de clave secreta. Una sola clave para cifrar y descifrar. DES, IDEA
 - **Cifrado asimétrico:** algoritmos de clave pública/privada. Dos claves por usuario(A), una pública Kpub(A) y otra privada Kpriv(A). Conocida Kpub(A) es imposible conocer Kpriv(A). Las claves son diferentes para cifrar y descifrar. RSA



WUOLAH

- **Autenticación con clave secreta:** “Ser quien se dice ser”. Tener garantía de la identidad de forma reciente
- **Intercambio de Diffie-Hellman (establecimiento de clave secreta):** permite establecer una clave secreta entre dos entidades a través de un canal no seguro. Ataque: man-in-the-middle
- **Funciones Hash. Hash Message Authentication Code (HMAC):** garantiza la integridad. Características:
 - Funciones unidireccionales (irreversibles) de cálculo sencillo
 - Texto de entrada (M) de longitud variable
 - $M \rightarrow H(M)$ siendo $H(M)$ de longitud fija (256 ó 512 bits)
 - Imposible obtener M a partir de su resumen $H(M)$
 - Invulnerables a ataques de colisión, dado M es imposible encontrar $M' \neq M$ tal que $H(M') = H(M)$
 - Ejemplos de funciones HASH: MD5, SHA-1, SHA-512
 - Las funciones Hash se usan para garantizar integridad + autenticación Hash Message Authentication Code(HMAC): $M + H(K|M)$ pero para evitar ataques de extensión se usa $M + H(K | H(K | M))$
- **Firma Digital:** sus objetivos son que el receptor pueda autenticar al emisor, que no haya repudio y que el emisor tenga garantías de no falsificación (integridad)
 - **Firma digital con clave secreta:** Big Brother
 - **Firma digital con clave asimétrica:** doble cifrado, uno para proporcionar privacidad, con K_{pubB} , otro, previo, para autenticación, con K_{priA} . Para firmar, enviar $K_{pubB}(K_{priA}(T)) \rightarrow$ en el receptor $K_{pubA}(K_{priB}(K_{pubB}(K_{priA}(T)))) = T$. Su debilidad: para garantizar el no repudio se necesita garantizar la asociación fehaciente e indisoluble de la “identidad A” con su “clave pública K_{pubA} ” ($A \leftrightarrow K_{pubA}$) ... ? esto se consigue con un “certificado digital”
- **Certificados digitales:** Para garantizar la asociación “identidad-clave”
 - Autoridades de certificación (AC): AC = Entidad para garantizar la asociación entre identidad y claves:
 - + El usuario obtiene sus claves pública y privada
 - + Éste envía una solicitud, firmada digitalmente, a la AC indicando su identidad y su clave pública
 - + AC comprueba la firma y emite el certificado solicitado:
 - Identidad de AC, identidad del usuario, clave pública del usuario y otros datos como, por ejemplo, el período de validez del certificado
 - Todo ello se firma digitalmente con la clave privada de AC con objeto de que el certificado no pueda falsificarse
 - Formato de certificados: principalmente X.509

Implementación de Mecanismos de Seguridad

Los mecanismos de seguridad hay que implementarlos en todos los niveles.

- **Seguridad Perimetral:** Software que inspecciona el tráfico entrante y saliente + un conjunto de reglas que filtran dicho tráfico. Firewalls + sistemas de detección de intrusiones (IDS) y de respuesta (IRS)
- **Seguridad en Protocolos** (¿dónde poner la seguridad?):
 - Capa de aplicación:
 - Pretty Good Privacy (PGP) → correo electrónico seguro. Coge el texto y crea un HASH, lo cifra con su clave privada y lo comprime. El destino lo descifra con una clave cifrada con la clave pública del destino (clave de sesión).
 - Secure Shell (SSH) → acceso remoto seguro
 - Capa de transporte
 - Transport Secure Layer (TSL) (antes SSL). Familia de protocolos (SSL Record Protocol, SSL Handshake Protocol, SSL Assert protocol y Change Cipher Espec Protocol).
 - HTTPS, IMAPS, SSL-POP, VPN. TLS = Handshake (negociar) + Record Protocol (operación).
 - TLS → Confidencialidad (Ksecreta negociada) + Autenticación (para el server por defecto con KPUBLICA) + integridad (Con HMAC)
 - Capa de red → IPSec (VPN)
 - Capas inferiores → PAP, CHAP, MS_CHAP, EAP...

ISec

Su objetivo es garantizar autenticación, integridad y (opcionalmente) privacidad a nivel IP. IPsec son 3 procedimientos:

1. Establecimiento de una “Asociación de seguridad”: IKE =RFC 2409.
 - Objetivo: establecimiento de clave secreta (Diffie-Hellman)
 - Incluye previamente autenticación (con certificados) para evitar el ataque de persona en medio
 - Es simplex: la asociación de seguridad tiene un único sentido.
 - Se identifica con la IP origen + Security Parameter Index (32 bits)
 - Vulnera el carácter NO orientado a conexión de IP.
2. Garantizar la autenticación e integridad de los datos: protocolo de “Cabeceras de autenticación”, RFC 2401
3. (Opcional) Garantizar la autenticación e integridad y privacidad de los datos: protocolo de “Encapsulado de seguridad de la carga”, RFC 241

IPSec tiene 2 modos de operación

- Modo Transporte: la asociación se hace de extremo a extremo entre en host origen y host destino
- Modo túnel: la asociación se hace entre dos routers intermedios. Ruta parcial de la ruta original

2.6 Aplicaciones Multimedia

Aplicaciones Multimedia: audio, vídeo, juegos, real-time

Calidad de servicio (QoS): capacidad de ofrecer el rendimiento requerido para una aplicación

IP ofrece Mejor esfuerzo (best effort): sin garantías de QoS

Tipos de aplicaciones:

- Flujo de audio y vídeo (streaming) almacenado → Ej. YouTube
- Flujo de audio y vídeo en vivo → Ej. emisoras de radio o IPTV
- Audio y vídeo interactivo → Ej. Skype

Características fundamentales

- Elevado ancho de banda
- Tolerantes relativamente a la pérdida de datos
- Exigen Delay (retardo) acotado
- Exigen Jitter (fluctuación del retardo) acotado
- Se pueden beneficiar de usar de multicast (direcciones destino de grupo)

2.7 Aplicaciones para Interconectividad de Redes Locales

Para asignar las direcciones se usa: DHCP (Dynamic Host Configuration Protocol), protocolo usuario de UDP (puerto 67).

- El host (cliente) envía un mensaje broadcast “DHCP discover”
- El server DHCP responde con un mensaje “DHCP offer”
- El host solicita una dirección IP, mensaje “DHCP request”
- El server DHCP envía la dirección IP: mensaje “DHCP ack”

Descubre la gama ZBook para creativos.

En Septiembre con 6% dto. adicional sólo para ti. Usa el cupón: WUOLAH6

Últimos días del cupón



Tema 3

Capa De Transporte En Internet

3.1 Introducción

La capa de transporte en TCP/IP es la primera de las capas denominada extremo a extremo. Esto quiere decir, que el diseño de esta capa consiste en situar la complejidad en los extremos, en las estaciones finales o hosts.

Funciones y servicios de la capa de transporte:

- **Ofrece una comunicación extremo a extremo:** involucra directamente a las estaciones finales o hosts y no a los dispositivos intermedios de la subred. Sitúa la complejidad en los extremos
- **Realiza la multiplexación/demultiplexación de aplicaciones -> puerto**

Para la capa de transporte se adopta una aproximación polarizada en la que se especifican dos alternativas:

- UDP:
 - Ofrece un servicio no orientado a conexión, no fiable (sin recuperación de errores)
- TCP:
 - Ofrece un servicio orientado a conexión, fiable. Incluye:
 - Control de errores y flujo
 - Control de la conexión
 - Control de congestión

3.2 Protocolo de Datagrama de Usuario (UDP)

También denominado datagrama de usuario, datagrama UDP o paquete UDP, ofrece una funcionalidad denominada "best-effort" o de buena voluntad. La principal característica de este protocolo es que ofrece un servicio no fiable y no orientado a conexión. En definitiva, lo que se pretende es que la entrega sea tan rápida como se pueda (no hay control de gestión ni de flujo). Esto implica que no hay hand-shaking, ni retardos de establecimiento. Puede haber pérdidas y no hay garantía de entrega ordenada.

La responsabilidad de los problemas de confiabilidad, pérdida, duplicación, desordenación, retrasos de paquetes, así como la pérdida de conexión recae sobre la aplicación, no sobre el protocolo.

La única diferencia al servicio ofrecido por IP, es que UDP añade multiplexación. Esto es un esquema de direccionamiento extra, basado en puertos, que permite identificar las aplicaciones destino y origen del datagrama. Así, toda comunicación entre una aplicación origen y otra destino precisa especificar no solo las respectivas direcciones IP, sino también los puertos origen y destinos correspondientes dentro de cada host. Cada segmento UDP se encapsula en un datagrama IP.

Los puertos van desde el 0 hasta el 65535. Desde el 0 al 1023, son puertos reservados por el sistema, del 1024 al 49151, son puertos de usuario, y desde el 49152 al 65535 son puertos dinámicos y/o privados. Es decir, que a partir del 1024 están a libre disposición del desarrollador.

Los campos que lo componen son:

- **Puerto origen:** campo de 16 bits que especifica el puerto origen.
- **Puerto destino:** campo de 16 bits que especifica el puerto destino.
- **Longitud UDP:** 16 bits que indican el número de octetos de que consta el datagrama UDP completo.
- **Comprobación:** campo redundante que contiene la suma de comprobación usual, es decir, el complemento a uno de la suma en complemento a uno de todo el datagrama y de una pseudo-cabecera.
- **Datos:** contiene la PDU de capa superior.

Por último, debido a la sencillez de UDP, es la opción adecuada para aquellas aplicaciones que impliquen interacciones sencillas, como aplicaciones multimedia o juegos online en los que la fiabilidad no es un requisito indispensable. Se usa frecuentemente para aplicaciones multimedia, tolerantes a fallos y muy sensibles a los retardos (no admiten recuperaciones de errores), consultas DNS, aplicaciones en tiempo real, SNMP, tráfico broadcast/multicast

Cada segmento UDP se encapsula en un datagrama IP que nos permite encaminar, pero agrega la capacidad para distinguir entre varios destinos de un mismo sistema terminal. La responsabilidad de los problemas de confiabilidad, pérdida, duplicación, desordenación, retrasos de paquetes, así como la pérdida de conexión recae sobre la aplicación, no sobre el protocolo.

3.3 Protocolo de Control de Transmisión (TCP)

Las principales características de este protocolo son las siguientes:

1. Es **orientado a conexión**, permitiendo el envío secuencial de datos, es decir, estos se reciben en el mismo orden en que fueron transmitidos (envío orientado a flujo).
2. Es un **servicio punto a punto**. No sirve para comunicaciones multicast (de uno a muchos).
3. Es **full-duplex**. Hay un bus de datos y otro de lectura, para que se puedan hacer a la vez la lectura y la escritura, y en ambos sentidos.
4. **Garantiza la entrega ordenada** de las secuencias de bytes generadas por la aplicación “stream oriented” y se adapta a las condiciones de la red **dinámicamente**.
5. Es un **servicio extremo a extremo fiable** que incluye control de flujo y control de congestión con ventanas deslizantes con tamaño máximo adaptable.
6. **Incluye mecanismos de control de flujo** para la detección y recuperación de errores (ARQ) con confirmaciones positivas ACKs (acumulativas) y “timeouts” adaptables.
7. **Usa la técnica de incorporación de confirmaciones** (“piggybacking”), que hace referencia a la parte acumulativa de los ACKs.

8. Para mejorar su eficacia TCP se ADAPTA a las condiciones de la red DINÁMICAMENTE
9. Entrega ordenada de paquetes

TCP usa mecanismos de realimentación que pueden llegar a incrementar el retardo o la latencia de la aplicación que lo implemente. El uso de TCP es aconsejable para la transmisión remota de datos a larga distancia para aplicaciones que exijan un servicio fiable.

Funcionalidades:

- Multiplexación/desmultiplexación de diferentes aplicaciones origen y destino
- Control de la conexión (establecimiento y cierre): dada la naturaleza de TCP, este incluye mecanismos para el establecimiento y el cierre de la conexión con carácter previo y posterior, respectivamente, a la transmisión de los datos.
- Control de errores y de flujo: recepción correcta y ordenada del flujo de mensaje, por parte de la aplicación destino tal y como se generó en la aplicación origen. Es más, se encarga de soslayar las diferencias que haya en cuanto a la tasa de generación y consumo de información entre las dos aplicaciones involucradas, liberando a estas de la necesidad de ocuparse de este problema.
- Control de congestión: reduce el posible agotamiento de los recursos disponibles en la subred (ancho de banda de las líneas y de la capacidad de almacenamiento temporal en los routers)

Las TPDUs de TCP se denominan segmentos TCP. Cada segmento TCP se encapsula en un paquete (denominado datagrama) IP.

En la cabecera tenemos:

- Puerto origen y puerto destino.
- Número de secuencia.
- Número de acuse de recibo. Con este campo y el anterior se puede saber cómo, cuándo y qué mensaje se está enviando.
- Longitud de la cabecera TCP.
- Plazo de confirmación.
- Conjunto de bytes para establecer datos prioritarios; controlar los ACKs, el tamaño de la ventana en el receptor, para saber cuántos bytes se pueden enviar.
- Punteros por si se utilizan para controlar los datos urgentes.
- Campo opcional.
- Datos.

Campos importantes:

- **Hlen:** Campo de 4 bits de valor mínimo igual a 5, utilizado para indicar la longitud de la cabecera TCP.
- **Reservado:** Campo de 3 bits sin uso específico (normalmente con valor 000).
- **Datos:** Campo en el que se transporta el mensaje generado por la aplicación. Aunque la longitud de este campo no se especifica explícitamente, se puede calcular a partir de la siguiente operación: tlenIP - hlenIP*4 - hlenTCP*4, donde tlenIP es el

número total de octetos del datagrama IP (cabecera IP + segmento TCP) y hlenIP y hlenTCP las longitudes en palabras de 32 octetos, de la cabecera IP y de la cabecera TCP.

- **Opciones:** Este campo de longitud variable puede presentar dos formatos distintos
 - a. Un único octeto que define el tipo de opción.
 - b. Un octeto que indica la longitud de los datos correspondientes a la opción, seguida de los datos como tales. Inicialmente solo existen especificados tres tipos de opciones:
 - 0 → fin de la lista de opciones
 - 1 → sin operación, utilizado como relleno para hacer que la longitud de la cabecera del segmento TCP sea múltiplo de 32 bits.
 - 2 → tamaño máximo del segmento (MSS, <<Maximum Segment Size>>)

Multiplexación/Demultiplexación

El objetivo es transportar las TPDUs (segmentos TCP) al proceso correcto. Para ello se usan los puertos (números enteros de 2 bytes que identifican al proceso origen y al proceso destino).

Permite que varias aplicaciones ejecutadas en un mismo host (origen y destino) puedan comunicarse entre sí usando TCP. Para dirigir las aplicaciones concretas, los campos de 16 bits (2 bytes) puerto origen y puerto destino del segmento TCP especifican, respectivamente, los puertos origen y destino de la comunicación y, en definitiva, identifican el proceso generador y consumidor de la información transportada. Existen puertos preasignados con servicios normalizados:

Puerto	Aplicación/Servicio	Descripción
20	FTP-DATA	Transferencia de ficheros: datos
21	FTP	Transferencia de ficheros: control
22	SSH	Terminal Seguro
23	TELNET	Acceso remoto
25	SMTP	Correo electrónico
53	DNS	Servicio de nombres de dominio
80	HTTP	Acceso hipertexto (web)
110	POP3	Descarga de correo

Eco, daytime o servicio de nombres(DNS), pueden usar tanto UDP como TCP. Son servicios independientes del protocolo de transporte, dispondrán de un proceso accesible en el mismo puerto, es decir, son servicios multiprotocolo.

Otros puertos (>1024) están a libre disposición del desarrollador.

Cada “conexión TCP” se identifica por: puerto e IP origen y puerto e IP destino

Descubre la gama ZBook para creativos.

En Septiembre con 6% dto. adicional sólo para ti. Usa el cupón: WUOLAH6

Últimos días del cupón



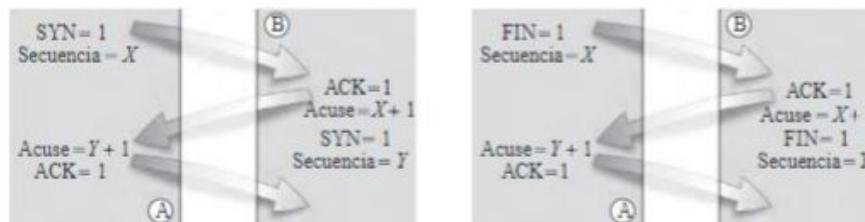
Control de la Conexión

Como es un protocolo orientado a conexión, previo al intercambio de los datos es necesario establecer una conexión entre las dos aplicaciones finales. Como es lógico, una vez finalizada la transmisión se cerrará la conexión. Se constituye de tres fases:

1. **Establecimiento de la conexión:** Uno de los extremos envía un segmento activado, indicando el deseo de establecer una conexión. En caso de aceptar la conexión, el otro extremo reserva recursos en el SO remoto. De igual forma, el otro extremo hace una solicitud al extremo origen, debido a la naturaleza full-duplex.

El extremo origen envía un valor X junto con SYN=1 (indicando que se quiere establecer una conexión) arbitrario a partir del cual se enumeran los segmentos. El otro extremo activa el bit ACK y hace acuse = X+1. Como el otro extremo tiene que hacer la transmisión en sentido contrario, envía un valor Y al extremo origen y este de nuevo activa el bit ACK y hace acuse = Y+1.

2. **Intercambio de datos (full-duplex)**
3. **Cierre de la conexión:** Es análogo al seguido en el punto 1, salvo que ahora cuando una de las dos entidades no tenga más datos que transmitir, enviará un segmento TCP con FIN=1 y de nuevo un valor X arbitrario.



Establecimiento (izquierda) y cierre (derecha)

¿Es posible garantizar un establecimiento/cierre fiable de la conexión sobre un servicio (IP) no fiable? **NO.**

Establecimiento de la conexión: three-way handshake

“Apertura activa”: cliente

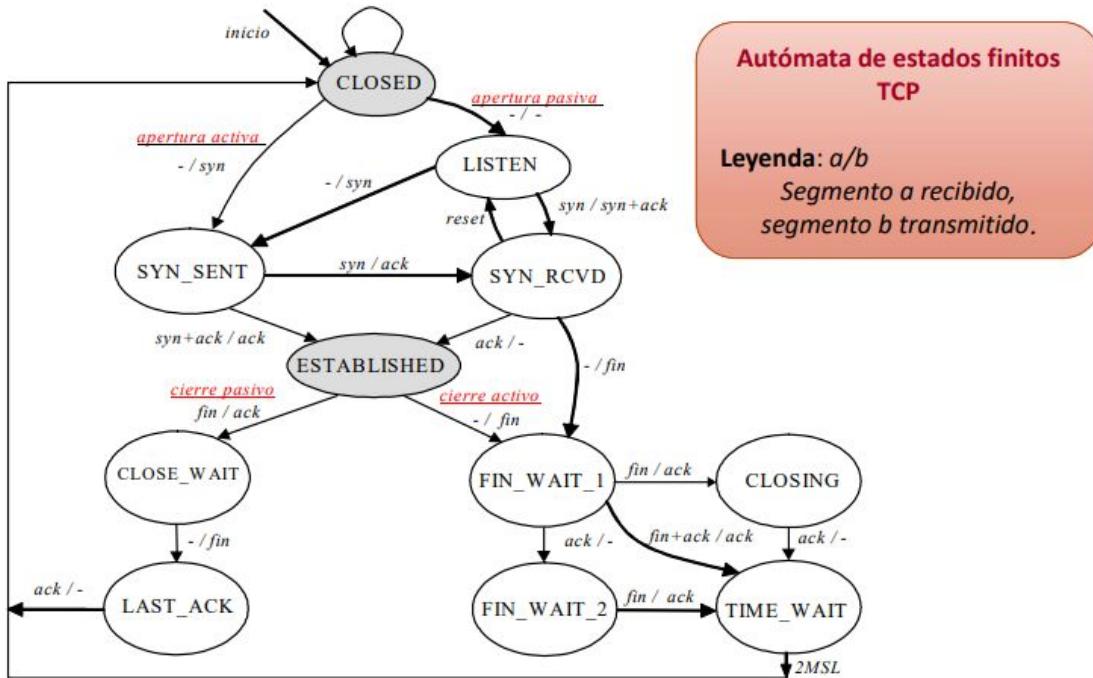
“Apertura pasiva”: servidor

El MSS (tamaño máximo del segmento), es la cantidad de datos (bytes) que se pueden transportar en un segmento. Cuanto mayor MSS, la utilización de la capacidad del canal será mayor, si bien lo deseable es elegir el MSS tal que evite la fragmentación del correspondiente paquete IP resultante. Es necesario conocer la menor MTU de todas las subredes implicadas en la ruta del destino.



WUOLAH

Es posible reiniciar el control de la conexión mediante activación del cuarto bit (R, de Reset) del campo control del segmento TCP. Es el segmento reinicio (Reset), el cual permite la desconexión ante situaciones anormales.



Campos involucrados:

- Bit S (SYN) del campo control
- Campo secuencia (valor aleatorio)
- Campo acuse
- Bit A (ACK) del campo control

Números de secuencia:

- El número de secuencia es un campo de 32 bits que cuenta bytes en módulo 2^{32} (el contador se da la vuelta cuando llega al valor máximo).
- El número de secuencia no empieza normalmente en 0, sino en un valor denominado ISN (Initial Sequence Number) elegido “teóricamente” al azar; para evitar confusiones con solicitudes anteriores.
- El ISN es elegido por el sistema (cliente o servidor). El estándar sugiere utilizar un contador entero incrementado en 1 cada 4 μs aproximadamente. En este caso el contador se da la vuelta (y el ISN reaparece) al cabo de 4 horas 46 min.
- El mecanismo de selección de los ISN es suficientemente fiable para proteger de coincidencias, pero no es un mecanismo de protección frente a sabotajes. Es muy fácil averiguar el ISN de una conexión e interceptarla suplantando a alguno de los dos participantes.
- TCP incrementa el número de secuencia de cada segmento según los bytes que tenía el segmento anterior, con una sola excepción: los flags SYN y FIN incrementan en 1 el número de secuencia.
- Los segmentos ACK (sin datos) no incrementan el número de secuencia

Control de Errores

El control de errores se basa en un esquema de realimentación con confirmaciones positivas y acumulativas. Es decir, las unidades de datos o segmentos que se envían han de ser confirmadas positivamente por parte del receptor.

Dado su carácter **full-duplex**, permite utilizar los segmentos de datos para incorporar en el campo acuse las confirmaciones de datos recibidos en sentido contrario. Esta técnica se llama piggybacking y permite ahorrar ancho de banda ya que no es necesario generar segmentos específicos de confirmación, si no que se aprovechan los segmentos generados en sentido contrario.

El esquema usado es ARQ con temporizadores y ventanas deslizantes de emisión y recepción:

1. Cada segmento se numera por parte del emisor y se envía. A la misma vez, se almacena una copia del segmento en la ventana de emisión y se inicia un temporizador asociado hasta que se recibe confirmación positiva del segmento, lo que se traduce por realizada la transmisión y se desplaza la ventana de emisión si procede.
2. En el receptor se mira el campo comprobación y se comprueba que el número de secuencia recibido está dentro de los definidos y se devuelve la confirmación positiva.
3. Si el temporizador llega a un valor determinado (timeout), se concluye que ha habido un error y se produce al reenvío de los datos.

Los campos que intervienen en el control de errores son:

- **Secuencia:** Offset en bytes dentro del mensaje
- **Comprobación:** Campo que se incluye en cada segmento transmitido y que permite al receptor determinar si el segmento se ha recibido correctamente o no.
- **Acuse:** Número de byte esperado en el receptor.
- **Bit A (ACK):** Indica la validez o no de la confirmación especificada en el campo acuse.

La eficacia del procedimiento dependerá de la correcta estimación del timeout. Si es muy pequeño, se desperdician recursos de red, y si es muy grande el emisor tardaría mucho en darse cuenta de posibles problemas. El valor de timeout debería estar próximo al tiempo de ida y vuelta entre el origen y el destino (esto se llama RTT, Round Trip Time).

Es mejor adoptar un timeout adaptable. La mejor solución es usar el Algoritmo de Karn, que dice que la estimación del timeout no se actualiza para segmentos que sean retransmisiones (es decir, si se produce un timeout, el segmento se reenvía, pues en ese momento no se actualiza el timeout). Surge así la técnica retroceso del temporizador (timer backoff), en la cual, cada vez que expira el temporizador asociado a un segmento TCP, se duplica el timeout viejo.

Control de Flujo

Es el procedimiento para evitar que el emisor sature al receptor con el envío de demasiada información y/o demasiado rápido. Para ello se utiliza el campo ventana del segmento, que indica el número de bytes que el receptor autoriza al emisor para que este los envíe de forma consecutiva.

El receptor especifica un tamaño de ventana máximo (ventana ofertada).

El emisor transmite datos con lo que se conoce como ventana útil, esto es, ventana útil = ventana ofertada - bytes en tránsito. Los bytes en tránsito son los ya transmitidos al receptor y de los cuales no se ha recibido confirmación.

Un posible problema ocurre cuando los segmentos se van haciendo cada vez más pequeños debido a la lentitud del emisor, del receptor o de ambos. Se conoce como el síndrome de la ventana tonta.

La solución consiste en hacer ventana útil más grande de lo que se oferta por el receptor, es decir: ventana útil > ventana ofertada - bytes en tránsito. Esto se basa en la suposición de que la transmisión se va a desarrollar con éxito.

TCP permite la transferencia de datos urgentes, es decir, permite que ciertos bytes se entreguen a la aplicación sin respetar el orden con el que fueron generados.

Para ello se puede activar el bit U. También se puede solicitar una entrega inmediata de datos para evitar altos retardos, activando para ello el bit P.

Control de Congestión

Se pueden llegar a experimentar retardos crecientes debido a la insuficiencia de recursos de la subred. Es un problema diferente al control de flujo, ya que involucra a la red y a los sistemas finales.

Puede tener naturaleza adelante-atrás, aunque en IP no. También, los episodios de congestión se manifiestan en retrasos en las ACKs y/o pérdidas de segmentos, dependiendo del nivel de severidad del episodio.

La solución extremo a extremo consiste en limitar en la fuente de forma adaptable el tráfico generado.

Esta limitación se hace mediante una aproximación por el tamaño de la ventana de emisión.

¿Qué es el producto BandWidth-Delay (RTT)? ¿Por qué es importante?

Procedimiento de prueba y error

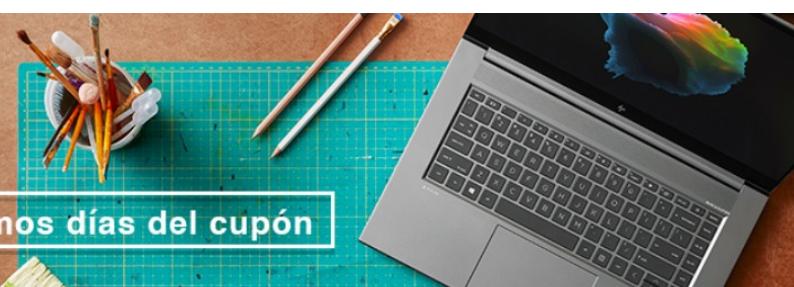
Se especificó un mecanismo de bucle cerrado basado en ventanas, con los siguientes parámetros:

- **Ventana_permitida:** Cantidad de datos que el emisor puede transmitir de forma consecutiva sin esperar confirmación

Descubre la gama ZBook para creativos.

En Septiembre con 6% dto. adicional sólo para ti. Usa el cupón: WUOLAH6

Últimos días del cupón



- **Ventana_del_receptor:** Cantidad de datos que el receptor autoriza a enviar al emisor.
- **Ventana_de_congestión:** Cantidad de datos que el procedimiento de control de congestión permite transmitir al emisor.

```
Bytes_permitidos_enviar =  
min{VentanaCongestion, VentanaDelReceptor}
```

VentanaDelReceptor: utilizada para el control de flujo (de tamaño variable) según el campo "ventana" recibido (ver pp. 12)

VentanaCongestion:
Inicialmente VentanaCongestion = 1 · MMS

Inicio lento
Si $\text{VentanaCongestion} < \text{umbral}$, por cada ACK recibido
 $\text{VentanaCongestion} += \text{MMS}$ (**crecimiento exponencial**)

Prevención de la congestión
Si $\text{VentanaCongestion} > \text{umbral}$, cada vez que se recibe todos los ACKs pendientes
 $\text{VentanaCongestion} += \text{MMS}$ (**crecimiento lineal**)

Si hay timeout entonces
 $\text{umbral} = \text{VentanaCongestion}/2$ y $\text{VentanaCongestion} = \text{MMS}$

Temporizadores TCP

Los temporizadores se usan para evitar los problemas que pueden desencadenarse por pérdidas o retrasos no deseados en la entrega de los paquetes IP, que en definitiva son los responsables de transportar los segmentos TCP. Se necesitan los siguientes temporizadores:

1. **Temporizador de establecimiento de conexión:** se inicia en la entidad servidora que recibe una solicitud de conexión (segmento SYN) y se cancela cuando el servidor recibe el ACK. Su objetivo es limitar el intervalo máximo de tiempo durante el cual el servidor reserva recursos al cliente en espera de establecer la conexión. Su valor por defecto es 75s.
2. **Temporizador de retransmisión:** se inicia en el emisor cada vez que se envía un segmento, su límite máximo es adaptable y depende de un suavizado temporal de los RTT medidos.
3. **Temporizador de ACK retrasados:** se inicia en el receptor coincidiendo con la llegada de un segmento ordenado sin discontinuidad en el número de secuencia. Su objetivo es esperar un cierto tiempo hasta que se reciba otro segmento y confirmar ambos acumulativamente. Su valor inicial es 200ms, aumentando exponencialmente hasta 500ms.
4. **Temporizador de persistencia:** se usa para evitar posibles bloqueos que pudieran ocurrir cuando se anuncia un tamaño de ventana de control de flujo igual a 0. Al recibir este anuncio, el emisor inicia el temporizador de persistencia, el cual se reinicia cuando recibe un anuncio de ventana mayor que cero. Si alcanza su valor

máximo, se envía un segmento de prueba al otro extremo para comprobar de esta manera su accesibilidad.

5. **Temporizador de mantenimiento (keepalive):** sirve para descartar que aunque no se reciben datos del otro extremo la entidad remota sigue operativa. Su valor es configurable, si no se recibe confirmación del otro extremo, se concluye que la conexión ha terminado y se cierra.
6. **Temporizador FIN-WAIT_2:** se habilita cuando desde una entidad TCP se envía un segmento FIN. Su valor máximo determina el intervalo de tiempo durante el cual esta entidad se mantiene activa esperando a recibir segmentos de datos válidos desde el otro extremo. Cuando una conexión pasa de estado FIN-WAIT_1 a FIN-WAIT_2 la conexión no puede enviar más datos. Cuando expira el temporizador si no ha habido cambio en el estado correspondiente, la conexión se pierde, el protocolo no se queda permanentemente en el estado FIN-WAIT_2. El valor de este temporizador puede llegar hasta los 10 min.
7. **Temporizador TIME-WAIT:** sirve para asegurar que todos los paquetes pendientes se confirman tras alcanzarse el estado TIME-WAIT es decir, tras recibirse un segmento FIN. Su valor por defecto es igual a 2 veces el tiempo de vida máxima de un segmento (MSL) el cual puede valer 30s, 1 min o 2. Durante este periodo se impide establecer nuevas conexiones entre las mismas direcciones IP y puertos para evitar que paquetes de la conexión previa se acepten en conexiones posteriores.

3.4 Extensiones TCP

TCP fue diseñado en un principio para suprir las necesidades de las redes de la década de 1970, pero las redes han cambiado mucho desde entonces y por tanto se han ido introduciendo mejoras a TCP. Estas mejoras son:

Adaptación de TCP a redes actuales (RFC 1323):

- **Ventana Escalada:** Campo ventana en el segmento. Es la máxima cantidad de datos que un receptor TCP autoriza a transmitir a un emisor sin esperar confirmación. Al principio, al ser un campo de 16 bits, la cantidad era de $2^{16} - 1 = 65535$ bytes. Esto es escaso para redes muy rápidas, por lo que se pasó a soportar hasta $2^{14} * 2^{16}$ bytes = 1GB autorizados.
- **Estimación del RTT:** Para mejorar la estimación de un único segmento por ventana, se propone el cálculo del RTT para cada segmento enviado usando un sello de tiempo.
- **Protección contra números de secuencia ya recibidos (PAWS):** Hace uso del sello de tiempo para rechazar segmentos duplicados que podrían corromper una conexión TCP abierta. Un segmento se considera duplicado cuando se recibe con un sello de tiempo menor que alguno de los recientemente recibidos para la conexión actual.
- **Confirmaciones selectivas:** TCP usa un mecanismo de ACK acumulativos, pero existe un problema. Si hay una pérdida aislada de un segmento, y seguida de esa

pérdida hay un buen número de segmentos con éxito, no hay manera de evitar la retransmisión de todos esos segmentos. Para ello se implementaron las confirmaciones selectivas (SACK, Selective ACKnowledgment). Con esta opción, además del esquema convencional, un segmento puede confirmar selectivamente bloques de bytes aislados.

Tema 4

Redes De Transporte

4.1 Introducción

Funciones y servicios de la capa de red en TCP/IP

- El objetivo de la capa de red en Internet es la interconexión de redes, con independencia de la tecnología subyacente
- Comutación: acción de cursar tráfico entre los nodos de la red
- Encaminamiento: encontrar la mejor ruta hasta el destino
- En el modelo OSI el control de congestión se realiza en esta capa

Ejemplos de protocolos de red:

- X.25
- IP

4.2 Comutación

Acción de cursar tráfico para establecer o determinar un camino que permita transmitir información extremo a extremo. Hay dos tipos de técnicas de comutación básicas existentes:

- **Comutación de circuitos:** se establece una conexión previa a la transferencia de la información entre las estaciones finales origen y destino. Una vez establecida la conexión, todos los datos que forman el mensaje se transmiten de forma secuencial siguiendo la misma ruta o circuito. Una vez concluida la comunicación se procederá al cierre de la conexión.

Ventajas:

- La transmisión se realiza en tiempo real, adecuado para voz
- Uso permanente de recursos, el circuito se mantiene durante toda la sesión
- No hay contención, no hay contienda para acceder al medio
- El circuito es fijo, no hay decisiones de encaminamiento una vez establecido
- Simplicidad en la gestión de los nodos intermedios

Desventajas:

- Retraso en el inicio de la comunicación
 - En ocasiones uso no eficiente de recursos
 - El circuito no es fijo. No se reajusta la ruta de comunicación
-
- **Comutación de paquetes:** en este esquema el mensaje no se transfiere secuencialmente como una sola unidad sino en trozos denominados paquetes, los cuales se transmiten nodo a nodo hasta alcanzar el destino. Existen dos variantes dentro de esta técnica:

Descubre la gama ZBook para creativos.

En Septiembre con 6% dto. adicional sólo para ti. Usa el cupón: WUOLAH6

Últimos días del cupón



Comutación de paquetes (datagramas): no se establece una conexión origen-destino previa a la transmisión. Características:

- Envío en unidades de datos (paquetes) independientes
- No hay conexión
- En cada salto: almacenamiento y reenvío
- Cada paquete debe contener en su cabecera las direcciones origen y destino
- Los paquetes pueden seguir rutas diferentes y pueden llegar desordenados

Comutación de paquetes con circuitos virtuales: de forma similar a como sucede en comutación de circuitos, si se establece tal conexión. Características:

- Usado en redes ATM (tecnología en desuso para redes troncales)
- Orientado a conexión. Pasos: (i) Conexión, (ii) Transmisión, (iii) Desconexión
- No hay asignación de recursos como en comutación de circuitos.

4.3 El Protocolo IP

IPv4 está especificado en el RFC 791 (1349,2474,6864). Características:

- Protocolo para la interconexión de redes (también llamadas subredes)
- Resuelve el encaminamiento en Internet: encontrar la ruta para llegar al destino
- Protocolo salto a salto: involucra a hosts y routers
- Ofrece un servicio no orientado a conexión y no fiable
 - No hay negociación o "handshake", no hay una conexión lógica entre las entidades.
 - No existe control de errores, ni control de flujo, ni control de congestión.
- La unidad de datos (paquete) de IP se denomina datagrama = cabecera + datos.
- IP es un protocolo de máximo esfuerzo ("best-effort") o buena voluntad: los datagramas se pueden perder, duplicar, retrasar o llegar desordenados.
- IP gestiona la fragmentación: adaptar el tamaño del datagrama a las diferentes Maximum Transfer Units (MTUs) de las subredes necesarias hasta llegar al destino.
- Cada entidad IP se identifica por su dirección IP
- Internet adopta un direccionamiento jerárquico que simplifica las tablas de routing.
- Las direcciones IP (32 bits) tienen dos partes bien diferenciadas: un identificador de la subred y un identificador del dispositivo (host) dentro de esa subred.
- Cada subred tiene un identificador único en la intranet (para dir. privadas) o en internet (para públicas)
- Cada dispositivo tiene un identificador único en la subred.
- La máscara de red es un patrón de 1s que determina qué bits pertenecen al identificador de subred
 - Dirección IP -> 200.27.4.112 = 11001000.00011011.00000100.01110000
 - Máscara -> 255.255.255.0 = 11111111.11111111.11111111.00000000
 - La máscara se puede representar de forma compacta, por ejemplo 200.27.4.112/24
- Dada una IP, para obtener la dirección o identificador de la subred, se realiza una operación lógica &

Se puede considerar internet como un conjunto de subredes interconectadas



WUOLAH

¿Qué es una Subred?

Líneas de transmisión e infraestructuras de red que permiten la conexión DIRECTA de dispositivos IP sin intermediarios (routers) “Para determinar las subredes, separe cada interfaz de los hosts y routers, creando redes aisladas. Dichas redes aisladas se corresponden con las subredes.”

¿Quién Tiene Direcciones IP?

Los hosts y los routers tienen 1 dirección IP por cada interfaz. Los switches operan en cada 2 -> NO tienen direcciones IP

¿Cómo se Elige la Máscara?

Según el número de dispositivos previsibles en la subred, tal que se ajusta para no desaprovechar direcciones. Recuérdese: cada subred tiene un identificador único en nuestra intranet (direcciones privadas) o en internet (públicas).

Ej: Dirección IP -> 200.27.4.112 = 11001000.00011011.00000100.01110000

Máscara -> 255.255.255.0 = 11111111.11111111.11111111.00000000

¿Cuántos Dispositivos por Máscara?

dispositivos = $2^{\#}$ ceros - 2 -> ej. 8 ceros (/24) permite 254 dispositivos

Direcciones

Direcciones públicas (identificador único en internet)

- Cada dirección se asigna a sólo 1 dispositivo en toda la Internet global.
- Se asignan centralizadamente

Direcciones privadas (identificador único en la intranet)

- Sólo sirven para tráfico dentro de las intranets.
- Se pueden repetir en distintas intranets.
- Las asigna el usuario según su criterio.

Direcciones IPv4: CLASES (ver RFC 1166 y 5737)

- Los hosts y routers tienen una IP por cada una de sus interfaces.
- 32 bits, notación decimal con puntos. Ejemplo: 192.168.212.60
- Originariamente se definieron 5 clases de direcciones IP (A,B,C,D,E)
- Clases A,B,C -> Jerárquicas a dos niveles:
 - Identificador de subred + identificador de dispositivo (host)
- Clase D : para multicast
- Clase E: usos futuros

NAT (“Network Address Translation”) RFC 1631, 2663, 3022

- NAT es un método para reasignar un espacio de direcciones IP (típicamente privadas) a otro (públicas) modificando la dirección IP de los paquetes mientras se retransmiten a través de un router
- En su uso habitual, el router NAT reemplaza las direcciones IP privadas origen salientes por direcciones públicas, y al revés con las entrantes

- Para ello el NAT usa una “Tabla de Traducciones”, que con la ayuda de una reasignación de puertos, permite deshacer los cambios en el tráfico entrante.
- IMPORTANTE: No se suelen instalar servidores (detrás de un NAT) con direcciones privadas, pues no serían accesibles desde el exterior. Una posible solución es usar STUN (Session Traversal Utilities for NAT): protocolo cliente/servidor que permite a clientes NAT encontrar su dirección IP pública, el tipo de NAT en el que se encuentra y el puerto asociado en la tabla de NAT con el puerto local.

El Encaminamiento

Conjunto de decisiones que deben tomar a fin de establecer la ruta o rutas a seguir para llevar la información (paquete a paquete) entre una estación final origen y una destino dadas. Se realiza paquete a paquete y salto a salto, en función de la IP destino del paquete y de las tablas de encaminamiento residentes en cada una de las entidades IP (host origen y routers).

En cada salto IP hay un procedimiento de store & forward (Almacenamiento y Retransmisión).

- Modos de encaminamiento:
 - Directo
 - No directo
- Cada dispositivo (host o router) tiene una tabla de encaminamiento
- Un router suele estar en varias redes distintas, un host suele estar en solo una red
- Al consultar la tabla, en caso de conflicto, se elige la ruta con la máscara más larga
- El default no es obligatorio, pero simplifica mucho las tablas
- Si no hay fragmentación y no hay “traducción de direcciones” (NAT), el datagrama (salvo el TTL, las opciones y el campo de comprobación) no se modifica en el camino.
- Proceso de encaminamiento en los nodos IP (salto a salto) por cada datagrama:
 - Se extrae la dirección destino: IP_DESTINO del datagrama
 - Por cada entrada i de la tabla de encaminamiento, con $i = 1, \dots, N$, se calcula:

$$IP_i = IP_DESTINO \text{ AND}(\&) \text{ MASCARA}_i$$
 - Si $IP_i == Di$ (campo destino de la tabla de encaminamiento) y si es routing directo (*) -> reenviar el datagrama al destino final por la interfaz I_i o si no es routing directo -> reenviar el datagrama al “salto siguiente” por la I_i
 - Si hay varias coincidencias, se elige el destino Di con la máscara más larga
 - Si se ha barrido toda la tabla y no hay coincidencia con ninguna fila -> error (posible mensaje ICMP)
 - Para encapsular el datagrama en la trama física correspondiente, se debe consultar la tabla ARP (ver más adelante) y en caso de no conocer la dirección física se envía un broadcast con protocolo ARP para obtener la dirección física.

Para facilitar la administración y aumentar la escalabilidad Internet se jerarquiza en Sistemas Autónomos (SA).

- Un SA es un conjunto de redes y routers administrados por una única autoridad que define cómo es el intercambio de tablas (routing interno) dentro del SA

- En cada SA existe un router, denominado router exterior, responsable de informar a los otros SAs sobre las redes accesibles a través del SA
- Cada SA se identifica por un entero de 16 bits (DESDE 2007 ES 32-BITS). Por ejemplo Rediris = AS766

Intercambio automático de tablas de encaminamiento

Se definen 2 niveles para el intercambio de tablas:

- Protocolos IGP: el administrador tiene libertad para elegir el protocolo de intercambio de tablas entre los routers dentro del SA. Ejemplos de protocolos IGP: RIP, OSPF, HELLO, IS-IS, IGRP, EIGRP
- Protocolo EGP (norma única en Internet) para el intercambio de información entre SA. Todos los “routers exteriores” usan el protocolo único en Internet: BGP

RIP (“Routing Information Protocol” RFC 1058, 2453, 4822)

- Protocolo de la capa de aplicación (opera sobre UDP puerto 520).
- Adopta un algoritmo vector-distancia (métrica basada en número de saltos).
- Periódicamente (por defecto cada 30 segundos) cada router RIP recibe Y envía a todos sus vecinos (dirección multicast 224.0.0.9) los vectores- distancia para todos los posibles destinos.
- De entre ellos, para un destino dado en la tabla de encaminamiento, se selecciona como salto siguiente (gateway) el vecino que anuncie el menor coste a ese destino, actualizando la métrica para ese destino, sumando 1 al coste anunciado.
- Problema las malas noticias tardan en propagarse.
- Problema de la convergencia lenta ó cuenta al infinito.
- Soluciones: Split horizon Hold down Poison reverse • Ver > man routed (SO Linux)

4.4 Asociación con Capa de Enlace: el Protocolo ARP

Direcciones MAC

- Tras consultar la tabla de encaminamiento → enviar el datagrama a la dirección Medium Access Control (MAC) del siguiente nodo. Se usan en redes Ethernet (cableadas) y Wifi
- Formato de las MAC (6 bytes): HH-HH-HH-HH-HH-HH -> ej. 00-24-21-A8-F7-6A
- Son únicas, asignadas por IEEE en lotes de 224 para cada fabricante
- Existe definida una dirección de difusión (broadcast) FF-FF-FF-FF-FF-FF
- Protocolo: Address Resolution Protocol (ARP) Obtener la dir. MAC a partir de la IP:
(a) y (b)
- Protocolo: Reverse ARP (RARP) Obtener la IP a partir de la MAC: (a) y ©

Descubre la gama ZBook para creativos.

En Septiembre con 6% dto. adicional
sólo para ti. Usa el cupón: **WUOLAH6**

Últimos días del cupón



4.5 El Protocolo ICMP

- Informa sobre situaciones de error en IP → es un protocolo de señalización
- Suelen ir (excepto eco y solicitudes) hacia el origen del datagrama IP original
- ICMP se encapsulan en IP
- Cabecera de 32 bits
 - Tipo (8 bits): tipo de mensaje
 - Código (8 bits): subtipo de mensaje
 - Comprobación (16 bits)