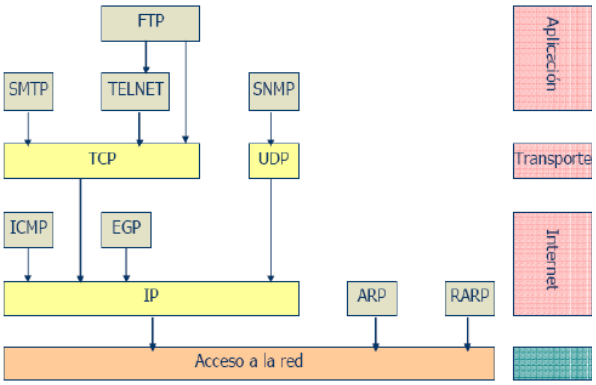
TEMA 2: SERVICIOS Y PROTOCOLOS DE APLICACIÓN EN INTERNET

1. Introducción a las aplicaciones de red

1.1 Arquitectura cliente servidor



Servidor

Está siempre en funcionamiento, esperando una petición del cliente. Tiene una **IP permanente y pública**. Se encuentran agrupados en "granjas".

Clientes

Su funcionamiento es intermitente. Pueden tener IP dinámica y privada. Se comunican con el servidor, pero no lo hacen entre sí.

1.2 Sockets

Un socket es un canal lógico que permite la conexión entre máquinas. También se puede definir como un tipo especial de manejador de fichero que utiliza un proceso para pedir servicios de red al SO. Un proceso cliente es el proceso que inicia la comunicación y un proceso servidor es el que espera a ser contactado. Los procesos envían y reciben mensajes a/desde su socket.

Para recibir mensajes de un proceso debe tener un identificador (IP + puerto). El **puerto** es un identificador (localizador interno al proceso) con el que se recuerda dónde está un proceso (de cara hacia afuera). A diferencia de un proceso de un SO, el identificador del puerto es fijo. El socket garantiza la comunicación y la seguridad (según la que ofrezca el protocolo).

1.3 Retardo en cola

El uso de un servidor se modela con un sistema M/M/1, es decir, la distribución de llegada de trabajos a la cola y la distribución de servicio ofrecida en un servidor indica que los tiempos entre eventos tienen una distribución exponencial (Poisson).

El retardo en cola es: $R = \frac{\lambda \cdot (T_s)^2}{1 - \lambda \cdot T_s}$ (T_s: tiempo de servicio, λ: ratio de llegada de solicitudes)

Esta misma expresión se puede utilizar para calcular el retardo en cola de un router.

1.4 Qué definen los protocolos de aplicación

Los protocolos de aplicación definen los **tipos de servicios**, **tipos de mensajes** (request, response, etc.), **sintaxis** (estructura de "campos" en el mensaje), **semántica** (significado de los "campos") y **reglas** (cuándo los procesos envían/responden mensajes).

Los tipos de protocolos existentes son: protocolos de dominio público (definidos en RFCs, como HTTP y SMTP), protocolos propietarios (Skype), in-band vs. out-of-band, stateless vs. state-full, persistentes vs. no persistentes.

La tendencia es hacer los protocolos flexibles con una cabecera fija y una serie de “trozos”, de los cuales algunos serán obligatorios y otros opcionales. Cada uno de esos “trozos” pueden incluir una cabecera específica más una serie de datos en forma de parámetros que comienzan en múltiplos de 4 bytes.

1.5 Características

- **Pérdida de datos:** algunas apps (audio, vídeo) pueden tolerar alguna pérdida de datos; otras (FTP, TELNET) requieren transferencia 100% fiable.
- **Requisitos temporales:** algunas apps (telefonía por internet, juegos interactivos) requieren bajo retraso (delay) para ser efectivas.
- **Rendimiento (throughput):** algunas apps requieren envío de datos a un ritmo determinado.
- **Seguridad:** encriptación, autenticación, no repudio, …

1.6 Protocolos de transporte

- **TCP:** servicio orientado a conexión, con transporte fiable, control de flujo y control de congestión
- **UDP:** no orientado a conexión(1), con transporte no fiable y sin control de flujo ni de congestión.

TCP y UDP (capa de transporte) al ser usuarios del protocolo IP (capa de red) no garantizan retardo acotado, fluctuaciones acotadas, mínimo rendimiento ni seguridad.

(1) La comunicación entre dos punto finales de una red en los que un mensaje puede ser enviado desde un punto final a otro sin previo acuerdo.

2. Servicio de nombre de dominio (DNS)

DNS (*Domain Name System*) realiza la traducción de nombres a direcciones IP, pues la comunicación en Internet precisa de este tipo de direcciones.

Respuesta del servidor

- **Respuesta con autoridad:** el servidor tiene autoridad sobre la zona en la que se encuentra el nombre solicitado y devuelve la dirección 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.

3. La navegación Web

Una página web es un fichero formado por objetos (ficheros HTML, imágenes JPEG, Java applets, ficheros de audio, etc.). Cada objeto se direcciona por una URL (http://servidor[:puerto]/path).

El **protocolo HTTP** sigue un modelo cliente-servidor, donde el cliente es un browser que pide, recibe y muestra objetos web y el servidor envía objetos web en respuesta a peticiones. La solicitud HTTP se hace siempre con la IP e intenta no cargar mucho al servidor.

Un protocolo es un lenguaje entre el cliente y el servidor con una sintaxis, una semántica y ciertas reglas de intercambio.

Características HTTP

Utiliza una conexión TCP (puesto que es más seguro que UDP, a pesar de que el delay sea mayor) al **puerto 80**: inicio de conexión TCP, envío HTPP y cierre de conexión TCP.

HTTP es **stateless** (no guarda estado), es decir, el servidor no nos reconoce, pues no mantiene información sobre las peticiones de los clientes; para ello utiliza cookies (el cliente guarda información sobre un servicio y manda la cookie con la petición).

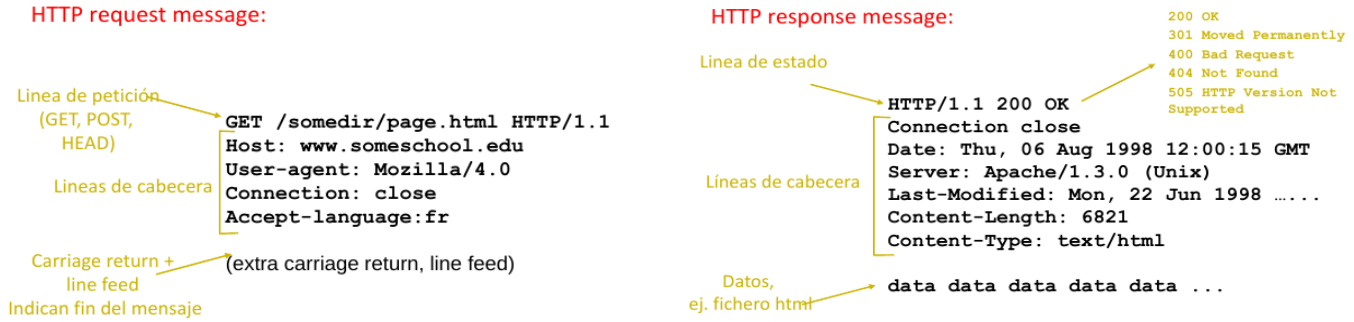
Según la negociación existen dos tipos de conexiones:

- **No persistente:** se envía únicamente un objeto en cada conexión TCP de manera secuencial.
- **Persistente:** pueden enviarse múltiples objetos sobre una única conexión TCP entre cliente y servidor, realizándose múltiples descargas en paralelo.

Un **proxy** es un servidor usado como intermediario para hacer análisis del tráfico, limitarlo, añadir una caché, etc.

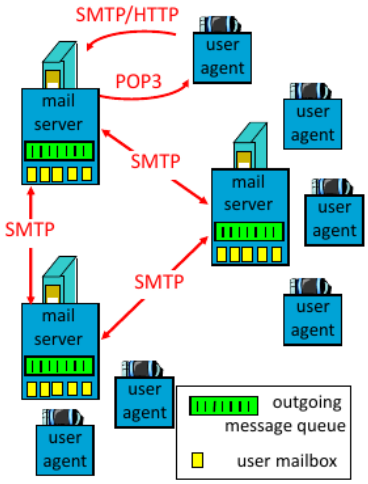
Tipos de mensajes HTTP

Existen dos tipos de mensaje HTTP: request y response



La **caché** busca satisfacer el requerimiento del cliente sin involucrar al servidor destino. Cuando un browser envía todos los requerimientos HTTP a la caché, si el objeto está en caché, ésta retorna el objeto, y si no, la caché requiere los objetos desde el servidor web y retorna el objeto al cliente. Una caché permite reducir la demanda hacia afuera y la latencia, de modo que no se compromete al servidor (excepto la primera vez).

4. Correo electrónico



Utiliza varios protocolos, siendo el principal SMTP (envío y comunicación entre servidores). Para descargar el correo se utiliza POP3 e IMAP. Alternativamente podría usarse HTTP, pero realmente no se estaría descargando, sino "leyendo" desde el servidor.

Tiene cuatro componentes principales:

- **Cliente de correo (user agent)**
- **Servidor de correo (mail server** o **mail transfer agent)**
- **Simple Mail Transfer Protocol (SMTP)**
- Protocolos de descarga: **POP3, IMAP, HTTP**

El *user agent* compone, edita y lee correos electrónicos. El servidor de correo almacena los mensajes salientes y entrantes de correo.

4.1 SMTP

Pasos para el envío/recepción de correo:

1. El usuario origen compone mediante su *user agent* un mensaje dirigido a la dirección de correo del usuario destino.
2. Se envía con SMTP o HTTP el mensaje al servidor de correo 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 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 *user agent* para leer el mensaje utilizando POP3, IMAP o HTTP.

SMTP se implementa mediante dos programas, **cliente SMTP** y **servidor SMTP**, que se ejecutan en el *mail server* que está enviando correo y recibiéndolo, respectivamente. Usa TCP y se lleva a cabo en tres fases: handshaking (saludo), transferencia de mensajes y cierre.

La interacción entre el cliente SMTP y el servidor SMTP se realiza mediante comandos (texto ASCII) y respuesta (código de estado y fases). Los mensajes deben estar codificados en ASCII de 7 bits.

4.2 Protocolos de acceso al correo

- **POP3:** Este protocolo consta de tres fases: fase de autorización, fase de transacción (cliente) y fase de actualización (servidor). Está orientado a descargar el correo, pero no tiene porqué hacerlo. Borra en el lado del cliente.
- **IMAP:** Se introdujo como mejora de POP3. Tiene ciertas ventajas como la organización en carpetas en el lado del servidor (MTA), manteniendo la información entre sesiones y permitiendo la descarga de partes de los mensajes. Es posible acceder con varios clientes (incluido POP en modo descarga/guardar).
- **WebMAIL:** Permite la organización total del servidor, accesible desde cualquier cliente con HTTP/HTTPS. No utiliza el *user agent*.

5. Seguridad y protocolos seguros

5.1 Primitivas de seguridad

- **Confidencialidad:** sólo accede a la información quien debe hacerlo. Está fuertemente ligado a la encriptación (para que nadie lea –confidencialidad– y nadie pueda modificar –integridad–).
- **Responsabilidad:**
  - Autenticación: los agentes de la comunicación son quienes dicen ser (usuarios).
  - No repudio: no se puede negar el autor de una determinada acción.
  - Control de accesos: garantía de identidad para el acceso (servidor).
- **Integridad:** la información no ha sido manipulada. Está integrada con la confidencialidad.
- **Disponibilidad:** los servicios están activos. Los ataques por denegación de servicios se producen al instalar como virus pequeño servidores en los clientes. Esta primitiva es diferente a las anteriores.

5.2 Mecanismos de seguridad

- **Cifrado simétrico:** C=K(P) y P=K(C)

  Emisor y receptor comparten una clave pública, que será la misma para cifrar y descifrar. El emisor, utilizando la clave pública, encripta el fichero mediante un algoritmo que también ha de ser público. El receptor, utilizando la misma clave pública, lo desencripta y lo lee. Se llama cifrado simétrico porque la clave está en ambos extremos. La ventaja que tiene es la rapidez, pero con la desventaja de tener que compartir la clave. Se utiliza AES.

- **Cifrado asimétrico:** $C=K^{+}(P)$ y $P=K^{-}(C)$
  El destinatario tiene una clave privada y otra pública, ésta última a disposición de todos los transmisores, a través de la cual se cifra la información a enviar para que nadie pueda leer el contenido. El receptor ha de utilizar la clave privada (sólo él la conoce) para poder descifrar. La clave pública y la privada están relacionadas. Ante la ventaja de tener una clave privada, la desventaja es la lentitud que este proceso supone. Solución:
  - Fase 1: Conexión con cifrado simétrico a través de la clave pública (en la negociación).
  - Fase 2: Intercambio de datos asimétrico con clave privada.
- **Message Authentication Code:** $M \mid F(M,N)$

  Se utiliza un hash para el mensaje, que ha sido cifrado simétricamente por el emisor para no modificar el hash, que será entonces descifrado por el receptor.
- **Firma digital:** $M \mid F(M,K^{-}) \rightarrow$ comprobación con K^{+}

  Invierte el cifrado asimétrico: se generan claves públicas y privadas, se codifica con la privada y la firma se hace pública para que cualquiera pueda verificar mediante la clave pública. El problema que plantea es asociar al individuo con su par de claves.
- **Certificado:** $(ID + K^{+}) \mid F((ID + K^{+}), K^{-CA})$

  Se basan en autoridades en las que confiar que asocian la identificación de un individuo con su clave pública. Para ello éste ha tenido que generar su clave privada ante dicha autoridad. Ante esto, al tener una firma digital y verificar con la clave pública, se tiene la verificación de identidad del firmante porque fue otorgada por la autoridad que concede los certificados de autenticación.

5.3 Seguridad

Seguridad (criptográfica) en protocolos

- **Capa de aplicación**
  - Pretty Good Privacy (PGP): tiene todos los mecanismos anteriores pero no se basa en una autoridad certificadora.
  - Secure Shell (SSH): conexión encriptada.
- **Capa de sesión** (entre aplicación y transporte)
  - Secure Socket Layer (SSL): se toman los protocolos desde cero y se securizan: HTTPS, IMAPS, SSL-POP,…
  - Transport Secure Layer (TLS): continuación de SSL con cierta compatibilidad hacia atrás. Mucho más seguro que SSL, pero no ampliamente utilizado por compatibilidad de uso con los clientes.
- **Capa de red:** IPSec (VPN): sirven para extender una red privada a través de una infraestructura pública. Para acceder se hace un “túnel” seguro a partir del cual la transmisión está encriptada para que la información no sea accesible desde fuera, y le confiere la seguridad necesaria.

Seguridad perimetral y gestión de riesgos

Seguridad estableciendo un perímetro y un guarda.

- Firewall: analiza los paquetes observando IPs y puertos (capas 3 y 4).
- Sistemas de detección de intrusiones (IDS) en red (NIDS) o host (HIDS): observa el comportamiento del sistema (CPU, red, etc.) para detectar intrusiones.
- Otros: antivirus, evaluación de vulnerabilidades, seguridad de aplicaciones, filtrado web, anti-spam, etc.

6. Aplicaciones multimedia

Las aplicaciones multimedia se basan en UDP ya que predomina la velocidad y no importa perder un frame en una conexión. La calidad del servicio (QoS) es la capacidad de ofrecer el rendimiento requerido para una aplicación. Siempre se quiere una gran calidad de servicio, pero Intenet (best effort) no es garantía de QoS.

Hay varios tipos de aplicaciones multimedia:

- Flujo de audio y vídeo (streaming) almacenado: importa la velocidad de descarga y la productividad. El jitter afecta a la calidad → YouTube
- Flujo de audio y vídeo en vivo: afecta la productividad, el jitter y el delay → Emisoras de radio en vivo, IPTV, Twich
- Audio y vídeo interactivo: el servicio deja de funcionar ante cualquier problema → Skype

Las características fundamentales para un buen funcionamiento son un **elevado ancho de banda** (a pesar de la compresión), **tolerancia a la pérdida de datos**, **delay acotado** (streaming > vivo > interactivo), **jitter**(1) **acotado** (~2% delay) y uso de **multicast** (enviar streaming –generalmente– a varios destinos (broadcast))**.**

(1) Se denomina jitter o fluctuación del retardo a la variabilidad temporal durante el envío de señales digitales, una ligera desviación de la exactitud de la señal de reloj. Suele considerarse como una señal de ruido no deseada y se denomina con este término a un cambio indeseado y abrupto de la propiedad de una señal.

7. Aplicaciones para interconectividad de redes locales

Es necesario conocer los puertos

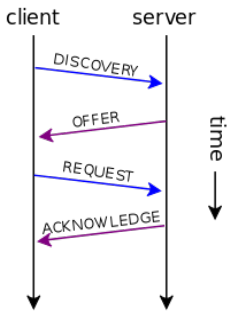
- **DHCP:** configuración dinámica de direcciones IP. Tiene los servicios en una base de datos y asigna automática/dinámicamente una dirección IP, además de otros parámetros de configuración de red. Se suele usar en las redes de hogar. Los servidores DHCP están en los routers. También se suele hacer DHCP en las conexiones a Internet.

  C: No se conoce la IP del servidor → El primer paquete de DHCP (DHCP Discovery) intenta conocer la IP del servidor; para ello hace un broadcast. Manda un ID.

  S: La respuesta es DHCP offer → Es un broadcast porque si hay varios servidores ofertando, el broadcast permite que el cliente pueda atrapar una respuesta. Contesta con un ID y un tiempo de vida que indica cuándo tiene que realizar otra solicitud el cliente.

  C: Vuelve a hacer broadcast para que todos los servidores sepan que se ha aceptado una oferta de uno de ellos (DHCP Request).

  S: Asigna la dirección IP (DHCP ACK).



- **DynDNS, No-IP, …:** servicios en la red privada, con IP pública variable. Configuración en router de acceso necesaria.
- **UPnP:** Abre puertos automáticamente (puede dar problemas de seguridad).