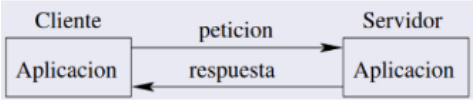
# Redes Tema 2. Capa de aplicación

Martín González Dios

22 de diciembre de 2024

Se ocupa de la **comunicación entre procesos**, para lo que usa protocolos de comunicación que permiten la comprensión de los mensajes y facilitan la programación de funciones de envío y recepción. Deben especificar el **tipo de mensaje** (petición, respuesta, ...), **reglas sobre el cuándo y cómo se envían los mensajes**, la **sintaxis del mensaje** (campos) y la semántica de cada campo. Usan **protocolos básicos de la capa de transporte y necesitan saber si usan TCP o UDP**, el ancho de banda y la temporización.

**Agente de usuario**: Interfaz entre usuario y aplicación (navegador, gestor de correo, ...).




## 1. HTTP

El **Protocolo de Transferencia de Hipertexto (HTTP)** define la comunicación entre un servidor web y un cliente web. Este protocolo utiliza el protocolo **TCP** y opera sobre el puerto **80**. Aunque HTTP es intrínsecamente *sin estado*, el uso de cookies permite dar la sensación de persistencia.

### 1.1. Estructura de una página web

Una página web consiste en un **documento HTML base**, que define la distribución y el texto, junto con varios **objetos** adicionales (archivos direccionables mediante URLs).

- El **navegador** actúa como el agente de usuario para la web.
- El **servidor web** alberga los objetos y responde a las solicitudes del cliente.
- Ante una solicitud de una página web, el servidor devuelve el documento base acompañado de sus objetos asociados.

### 1.2. Conexiones HTTP

Las conexiones HTTP pueden clasificarse en dos tipos:

- **Conexiones no persistentes:** Se utiliza una conexión TCP diferente para transferir cada objeto. Esto puede hacerse:
  - En serie: Cada objeto se transfiere esperando a que la conexión previa se cierre.
  - En paralelo: Varias conexiones se abren simultáneamente para transferir objetos.
- **Conexiones persistentes:** Se utiliza una única conexión TCP para transferir múltiples objetos e incluso páginas completas. Dentro de este tipo, encontramos:
  - **Sin entubamiento (sin pipelining):** El cliente solicita un nuevo objeto solo después de recibir el anterior.
  - **Con entubamiento (con pipelining):** El cliente puede realizar múltiples solicitudes de objetos sin esperar a recibir los anteriores.

## 1.3. Tiempo de transferencia de una página web

El tiempo de transferencia de una página web depende de:

- El **RTT** (Round-Trip Time), que es el tiempo de ida y vuelta cliente-servidor-cliente.
- El tiempo de **transmisión del archivo**, que depende del tamaño del archivo.

Para el primer paquete, el tiempo total es de $2 \times RTT + t_{\text{transmisión}}$. Para los paquetes restantes, el tiempo depende del tipo de conexión utilizada.

## 1.4. Mensajes HTTP

Los mensajes HTTP están compuestos por:

- **Cabecera:** Contiene información de control en formato ASCII (7 bits).
- **Cuerpo:** Contiene los datos, que pueden estar en formato binario.

Existen dos tipos de mensajes:

### 1.4.1. Mensajes de petición

- **Línea de petición:** Especifica el objeto solicitado. Ejemplos:
  - `GET`: Solicitud de una página normal.
  - `POST`: Envío de datos desde un formulario.
- **Líneas de cabecera (opcionales):** Ejemplos de campos incluyen:
  - `Host`
  - `Connection`
  - `User-Agent`
  - `Language`
- **Cuerpo:** Vacío para `GET` y con datos del formulario para `POST`.

### 1.4.2. Mensajes de respuesta

- **Línea de estado:** Indica el estado de la respuesta. Ejemplos de códigos:
  - `200`: Éxito.
  - `404`: No encontrado.
  - `400`: Solicitud incorrecta.
- **Líneas de cabecera (opcionales):** Ejemplos de campos incluyen:
  - `Connection`
  - `Date`
  - `Server`
  - `Last-Modified`
  - `Content-Length`
  - `Content-Type`
- **Cuerpo:** Contiene los datos de la respuesta.

## 1.5. HTTP/2

Con la llegada de HTTP/2 se introdujeron muchas mejoras respecto a la versión anterior:

- Multiplexación de múltiples peticiones HTTP sobre una misma conexión TCP.
- Compresión de cabeceras para reducir el tamaño de las transmisiones.
- Soporte para **server push**, que permite al servidor enviar recursos adicionales sin solicitud explícita del cliente.
- Pipelining de solicitud-respuesta.
- Cifrado obligatorio mediante SSL (Secure Sockets Layer).
- Incremento de velocidad: HTTP/2 es un 65 % más rápido que HTTP/1.1.

# 2. FTP

El **Protocolo de Transferencia de Archivos (FTP)** es un protocolo que define la comunicación entre un cliente y un servidor de archivos. Este protocolo es **con estado**, manteniendo la sesión activa durante toda la comunicación, y utiliza dos conexiones TCP paralelas para su funcionamiento.

## 2.1. Conexiones FTP

FTP emplea dos tipos de conexiones TCP:

- **Conexión de control:**
  - Utiliza el puerto **21**.
  - Es **persistente**, permaneciendo activa durante toda la sesión.
  - Se emplea para enviar comandos y recibir respuestas. Los comandos son cadenas de 4 caracteres en mayúscula, acompañadas de campos adicionales. Algunos ejemplos de comandos son:
    - `USER`: Especifica el nombre de usuario.
    - `PASS`: Especifica la contraseña.
    - `RETR`: Solicita la descarga de un archivo.
  - Las respuestas consisten en códigos de 3 dígitos seguidos de una frase explicativa. Algunos ejemplos de códigos de respuesta son:
    - `331`: Se requiere una contraseña.
    - `125`: La transferencia de datos está en progreso.
    - `425`: No se pudo abrir la conexión de datos.
  - Utiliza el formato ASCII de 7 bits para la comunicación.
- **Conexión de datos:**
  - Utiliza el puerto **20**.
  - Es **no persistente**, abriéndose una nueva conexión para cada archivo que se transmite.
  - Se emplea para transferir los datos en respuesta a los comandos recibidos a través de la conexión de control.
  - También utiliza el formato ASCII de 7 bits para la transferencia de datos.

## 2.2. Seguridad en FTP

Es importante destacar que FTP, al igual que Telnet y HTTP, no cifran las comunicaciones, lo que los hace vulnerables a intercepciones. Por el contrario, protocolos como SFTP, SSH y HTTPS cifran las conexiones, garantizando una mayor seguridad durante la transferencia de información.

# 3. Protocolos de correo electrónico

En el ámbito del correo electrónico, se utilizan diferentes protocolos para el envío y acceso a los mensajes. Los principales protocolos son **SMTP**, **POP3**, **IMAP**, y HTTP (cuando se utiliza un navegador para acceder al correo). Estos protocolos operan de la siguiente manera:

## 3.1. SMTP (Protocolo simple de transferencia de correo)

SMTP es el protocolo utilizado para enviar correo electrónico al servidor de correo o entre servidores. Este protocolo permite la comunicación entre el agente de usuario y el servidor de correo. Si el servidor de destino no está disponible, se realiza un reintento después de un intervalo de 30 minutos.

- **Puerto**: 25.
- **Conexión**: TCP persistente, es decir, se mantienen abiertas para varios mensajes.
- **Codificación**: Se utiliza ASCII de 7 bits para las cabeceras y los cuerpos de los mensajes.
- **Mensajes**: Existen tres tipos de mensajes en SMTP:
  - **Comandos**: Palabras en mayúsculas seguidas de parámetros, como `HELO`, `MAIL FROM`, entre otros.
  - **Respuestas**: Códigos numéricos seguidos de una frase, como `220`, `354`, `221`, etc.
  - **Datos**: El contenido de los correos electrónicos.
- **Seguridad**: SMTP es un protocolo inseguro ya que no requiere autenticación (no pide nombre de usuario ni contraseña), lo que facilita el envío de correos electrónicos no solicitados.
- **Naturaleza**: SMTP es un protocolo de oferta, en el que el cliente ofrece el mensaje al servidor, a diferencia de HTTP, que es de demanda. Además, SMTP requiere que los correos sean codificados en ASCII de 7 bits (a diferencia de HTTP que permite datos binarios).
- **Estado**: SMTP mantiene el estado de la sesión, lo que significa que recuerda la fase en la que se encuentra durante la transferencia de correos.

## 3.2. POP3 (Protocolo de oficina postal versión 3)

POP3 es un protocolo más sencillo que SMTP, utilizado para acceder a los correos electrónicos mediante una conexión TCP persistente. Opera sin mantener estado entre las sesiones.

- **Puerto**: 110.
- **Fases del protocolo**:
  - **Autorización**: En esta fase, se autentica al usuario con su nombre de usuario y contraseña.
  - **Transacción**: Se recuperan los mensajes del servidor, se marcan los correos para su borrado y se recopilan estadísticas sobre el correo.
  - **Actualización**: Se eliminan los mensajes marcados para borrar y se finaliza la sesión.
- **Comandos**: Los comandos en POP3 son palabras de 4 caracteres seguidas de parámetros, como `USER`, `PASS`, `LIST`, entre otros.
- **Respuestas**: Las respuestas del servidor son `+OK` (si la operación fue exitosa) o `-ERR` (si ocurrió un error).
- **Datos**: Los datos incluyen la lista de mensajes y el contenido de los correos. A diferencia de SMTP, POP3 entrega el correo completo en un solo mensaje, no de forma fragmentada.

### 3.3. IMAP (Protocolo de acceso a mensajes de Internet)

IMAP es un protocolo más avanzado que POP3. A diferencia de POP3, IMAP permite una mayor flexibilidad en el acceso y gestión de los correos electrónicos.

- **Carpetas**: IMAP asocia cada mensaje con una carpeta, lo que permite una organización más eficiente de los correos. POP3 no ofrece soporte para carpetas.
- **Comandos avanzados**: IMAP incluye comandos para crear carpetas, realizar búsquedas en los correos electrónicos, y más.
- **Estado entre sesiones**: IMAP mantiene el estado entre sesiones, lo que significa que conserva la información sobre la posición y el estado de los mensajes entre sesiones, algo que POP3 no permite.
- **Conexión**: Como POP3, IMAP utiliza una conexión TCP persistente, pero permite un acceso más completo y organizado al correo.

## 4. DNS (Servicio de Nombres de Dominio)

El DNS (Domain Name System) es un sistema que **traduce los nombres de host a direcciones IP y viceversa**. Además, tiene varias funciones importantes, como obtener alias, informar sobre los servidores autorizados para un dominio, proporcionar alias de servidores de correo y distribuir la carga mediante la asignación de varias direcciones IP a un nombre de host.

- **Funciones principales del DNS**:
  - Traduce nombres de hosts a direcciones IP.
  - Obtiene alias y proporciona alias de servidores de correo, lo que simplifica las direcciones.
  - Informa sobre los servidores autorizados para un dominio.
  - Distribuye la carga asignando varias direcciones IP a un solo nombre de host, utilizando servidores espejo y devolviendo cíclicamente una de las IPs asignadas.
- **Estructura del DNS**:
  - El sistema DNS está compuesto por servidores de nombres distribuidos a través de Internet.
  - Utiliza una base de datos jerárquica para almacenar la información.
  - Opera bajo un protocolo UDP sin estado, utilizando el puerto 53.
- **Tipos de servidores DNS**:
  - **Servidores locales**: Atienden las consultas realizadas por los hosts en la red local.
  - **Servidores autorizados o autoritativos**: Registran los hosts en dos servidores autorizados para que sean accesibles en Internet. Estos servidores pertenecen al ISP (Proveedor de Servicios de Internet) y muchos actúan como servidores locales.
  - **Servidores raíz**: Más de 400 servidores gestionados por 13 organizaciones. Tienen la información sobre los dominios de primer nivel (TLD).
  - **Servidores intermedios o TLD**: Contienen información sobre los niveles intermedios en la jerarquía de dominios.
- **Caché de DNS**:
  - En todos los niveles, los servidores DNS almacenan copias de las correspondencias que obtienen. Estos datos se borran después de cierto tiempo si no se utilizan.

### 4.1. Mensajes DNS

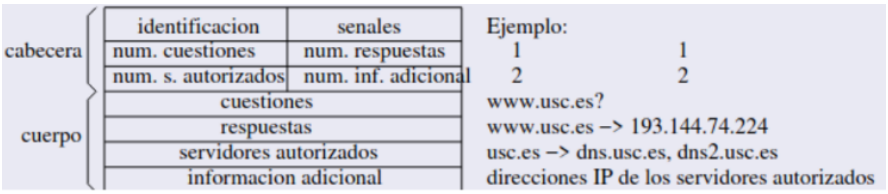
Los mensajes DNS se componen de una consulta y una respuesta, y están organizados de la siguiente manera:

- **Cabecera**: Contiene la información de control, incluyendo:
  - **Identificación**: 16 bits que identifican la consulta.
  - **Señales**: 4 bits que indican si se trata de una consulta o una respuesta y el tamaño de los campos.
- **Cuerpo**: Contiene las siguientes secciones:
  - **Cuestiones**: Información solicitada por la consulta.
  - **Respuestas**: Respuestas a la consulta.
  - **Servidores autorizados**: Información sobre los servidores DNS autorizados.
  - **Información adicional**: Datos adicionales que pueden ser útiles.

### 4.2. Tipos de consultas DNS

Las consultas DNS pueden ser de dos tipos:

- **Consultas recursivas**: En este tipo de consultas, cada servidor DNS pregunta al siguiente hasta obtener una respuesta final.
- **Consultas reiterativas**: En este caso, el servidor DNS local contacta con todos los servidores DNS involucrados, obteniendo respuestas directas de cada uno.




## 5. Distribución de Contenidos

El acceso a servidores centralizados puede ser lento debido a factores como caminos lentos, congestión de la red o servidores sobrecargados. Para mitigar esto, se distribuyen y duplican los contenidos en varias zonas, y las peticiones se dirigen al servidor que ofrezca el menor tiempo de respuesta.

### 5.1. Caché Web o Servidor Proxy

Un método común de distribución es la **caché web**, también conocida como **servidor proxy**. Este método es proporcionado por el ISP (Proveedor de Servicios de Internet) y se basa en un servidor intermedio (proxy) por el cual pasan las peticiones web de los hosts dentro de una red. El servidor proxy mantiene copias de los contenidos temporalmente, lo que mejora el tiempo de acceso a los recursos solicitados. Además, permite establecer un esquema jerárquico. Sin embargo, el usuario debe configurar su navegador para utilizar este método.

## 5.2. Redes de Distribución de Contenidos (CDN)

Otro enfoque de distribución es el uso de **Redes de Distribución de Contenidos** (**CDN**, por sus siglas en inglés). Este sistema se basa en empresas que poseen centros de hosting de Internet (compañías CDN) y alquilan su infraestructura. La CDN replica los contenidos de sus clientes en sus propios servidores y se asegura de que estos estén actualizados.

El contenido se entrega desde el **servidor CDN** que pueda ofrecer el acceso más rápido, lo cual mejora el rendimiento y la disponibilidad. El acceso a los contenidos puede ser realizado a través de **redirección de objetos** o mediante **balanceo de las peticiones**, utilizando el sistema DNS para dirigir las solicitudes al servidor más cercano o rápido.

## 5.3. Redes P2P (Peer-to-Peer)

Las **redes P2P** permiten que todos los usuarios actúen tanto como servidores como clientes mediante el uso de una aplicación. Este modelo elimina la necesidad de servidores centralizados, ya que los usuarios comparten directamente los contenidos entre sí.

Para su funcionamiento, una red P2P necesita un **nodo de arranque** y un **directorio** que puede ser centralizado, no centralizado o basado en la inundación de consultas.

### 5.3.1. BitTorrent

Con el protocolo **BitTorrent**, cada archivo se comparte dentro de una red P2P específica que no requiere un directorio centralizado. En lugar de ello, la información de los archivos se almacena en archivos **.torrent**, que contienen los detalles necesarios para que los usuarios puedan localizar y compartir los contenidos. Aunque no es necesario un directorio, el sistema BitTorrent sí necesita un **tracker** para cada red, el cual ayuda a coordinar las descargas y mantener la conexión entre los usuarios que comparten los archivos.