GENERALITAT VALENCIANA
Conselleria d'Educació, Cultura i Esport

ceedcv CENTRE ESPECÍFIC D'EDUCACIÓ A DISTÀNCIA DE LA COMUNITAT VALENCIANA

Unidad 3.1
Generación de servicios en red

Índice

Sumario

Licencia

Reconocimiento – NoComercial – CompartirIgual (by-nc-sa): No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

1. Introducción a la generación de servicios en red

Los servicios se podrían definir como unos programas auxiliares que gestionan recursos y proporcionan funcionalidades a usuarios y aplicaciones en un sistema. En esta unidad se estudiará la implementación de servicios y protocolos que facilitan la comunicación, el intercambio de datos y la administración de sistemas a través de redes informáticas. Se verá cómo opera cada uno de estos servicios que desempeñan un papel fundamental en la creación y mantenimiento de una infraestructura de red eficiente.

En el contexto de Internet, todos los servicios siguen una arquitectura cliente-servidor, donde la interacción entre clientes y servidores juega un papel fundamental. Los protocolos más destacados que estudiaremos en esta unidad serán el protocolo TELNET/SSH para la administración remota, FTP para la transferencia de archivos, SMTP para el envío de correos y POP3 e IMAP para la gestión de correos electrónicos.

Más allá de estos servicios específicos, existen otras aproximaciones basadas en la arquitectura cliente-servidor que ofrecen acceso a sistemas. Una de las más utilizadas es la API Rest, que opera sobre HTTP y permite la interacción con el backend de sistemas de forma independiente a la arquitectura y lenguajes utilizados en el frontend. Esta unidad se adentrará en estas dinámicas, analizando cómo estas tecnologías contribuyen a la conectividad y eficiencia en el ámbito de las redes informáticas.

2. TELNET/SSH

Aunque hoy está prácticamente obsoleto, Telnet fue un servicio de administración remota que permitía conectarse a máquinas UNIX desde un sitio remoto, permitiendo enviar comandos y ver el resultado de dichos comandos en nuestro terminal como si estuviésemos sentados en la máquina administrada.

El gran problema de Telnet fue la seguridad. Telnet no enviaba datos cifrados por lo que cualquier persona con un sniffer podía capturar el tráfico de una sesión y ver no solamente los comandos sino también los usuarios y contraseñas enviados a través de la red. Por ello, Telnet prácticamente no se usa hoy en día y ha sido sustituido «de facto» por SSH, que hace lo mismo pero cifrando la comunicación con criptografía de clave pública.

3. FTP

FTP significa «File Transfer Protocol» y es un protocolo orientado a comandos pensado para descargar ficheros. Podría decirse que también tiende a desaparecer, ya que cada vez más a menudo se usa el navegador (con HTTP) para descargar ficheros. En realidad, algunos navegadores, como Mozilla Firefox siguen implementando el protocolo FTP, permitiendo así el descargar ficheros como si en realidad estuviésemos usando comandos.

Cabe destacar que FTP tiene dos modalidades de uso: pasivo y activo. La diferencia entre activo y pasivo está en quien inicia las conexiones. Por defecto, los router no suelen permitir la entrada de conexiones iniciadas desde el exterior (aunque sí permiten que entren datos del exterior si desde el interior hemos iniciado previamente la conexión). Cuando se definió FTP como protocolo, se daba por sentado que todo el mundo aceptaría conexiones siempre, por lo cual en el FTP normal (el activo) se permite que el servidor inicie una conexión con nosotros para enviarnos un fichero. En FTP hay dos conexiones, una para enviar comandos (que inician los cliente) y otra (que inicia el servidor) para recibir el archivo. Podría decirse que en el FTP normal/activo «no se descarga» sino que «el servidor envía un fichero».

Cuando la seguridad empezó a preocupar a todo el mundo los fabricantes y compañías telefónicas empezaron a distribuir routers en los que por defecto no se aceptaban conexiones empezadas fuera de nuestra red, así que de repente el FTP empezó mostrar un comportamiento peculiar que podría resumirse en la frase «se pueden enviar comandos pero no descargar ficheros». Ese error tiene su lógica porque la conexión de comandos la inicia el cliente pero el servidor inicia la conexión (mejor dicho, lo intenta) para enviar el fichero al cliente. Debido a estos problemas se desarrolló el «modo pasivo», que consiste en que ahora el cliente inicia una conexión para enviar comandos e inicia otra conexión para descargar el fichero. Como la conexión de descarga se inicia «dentro de la red», el router autoriza la salida, toma nota de la conexión y cuando llega la respuesta, la deja pasar (porque observa que la conexión no se ha iniciado fuera, sino que es la respuesta a una conexión iniciada dentro). Como puede deducirse se llama pasivo porque el servidor ya no es activo, sino que se limita a recibir conexiones.

4. SMTP (Simple Mail Transfer Protocol)

"Simple Mail Transfer Protocol," es un protocolo utilizado para enviar correos electrónicos a través de una red. Funciona como un servicio de envío de mensajes, permitiendo que un servidor SMTP entregue los correos electrónicos salientes desde un cliente de correo electrónico hasta el servidor de destino o entre servidores SMTP. Este protocolo se centra en el envío de mensajes y no en la recepción ni almacenamiento. SMTP opera en el puerto 25 y es fundamental para el intercambio eficiente y confiable de correos electrónicos en Internet.

5. POP3 e IMAP

SMTP es el protocolo que se encarga del envío de correos electrónicos. En cambio POP3 e IMPA permiten la gestión de correos electrónicos. POP3 se caracteriza porque facilita la descarga y almacenamiento local, mientras que IMAP permite la gestión de correos electrónicos directamente en el servidor, ofreciendo mayor flexibilidad en entornos donde se accede a la misma cuenta desde diferentes dispositivos.

5.1. POP3 (Post Office Protocol, versión 3)

POP3, cuyas siglas corresponden a "Post Office Protocol," es un protocolo de acceso a correos electrónicos que permite a los usuarios recuperar mensajes desde un servidor de correo. La versión más común es POP3, la tercera iteración del protocolo. A diferencia de IMAP, POP3 descarga los correos electrónicos desde el servidor a la computadora del usuario y, por defecto, los elimina del servidor después de la descarga. Esto significa que los correos electrónicos están almacenados localmente en el dispositivo del usuario, y la gestión principal se realiza en el cliente de correo.

5.2. IMAP (Internet Message Access Protocol)

IMAP, que se traduce como "Internet Message Access Protocol," también es un protocolo de acceso a correos electrónicos, pero difiere de POP3 en varios aspectos. IMAP permite a los usuarios visualizar y manipular los correos electrónicos directamente en el servidor, en lugar de descargarlos localmente. Esto facilita la gestión de correos electrónicos desde múltiples dispositivos, ya que los cambios se reflejan en todos ellos. IMAP opera en el puerto 143 y se utiliza comúnmente cuando se busca acceder a correos electrónicos desde varios dispositivos de manera sincronizada.

6. HTTP

El protocolo para la transferencia de hipertexto (HyperText Transfer Protocol) es un protocolo que en su momento se diseñó para que los navegadores (clientes web) se conectarán a servidores y descarguen archivos HTML. Sin embargo, su uso se ha popularizado en otros ámbitos como son la creación de aplicaciones web, es decir aplicaciones pensadas para ser manejadas desde un navegador.

HTTP como protocolo de navegación

Todas las páginas web utilizan HTTP para gestionar la transferencia de diversos contenidos, como páginas con hipertexto (HTML), imágenes, vídeos y documentos.

- **Los servidores web más usados:**
  - Según W3Techs (mayo de 2022):Nginx, Apache y Cloudflare Server
  - Según Netcraft (mayo de 2022): Nginx, Apache y OpenResty
- **Los navegadores o browsers más usados son:**
  - Según statcounter (diciembre 2022): Chrome (64.62%), Safari (18.29%), Edge (4.24%), Samsung Internet (3.06%), Firefox (3.02%) y Opera (2.24%)
  - En Europa: Chrome (58.5%), Safari (21.63%), Edge (5.83%), Firefox (5.23%), Samsung Internet (3.37%) y Opera (3.35%)

Características del protocolo HTTP

El protocolo HTTP se destaca por emplear **el puerto 80** como su canal predeterminado para la comunicación.

HTTP es un protocolo basado en texto plano. Esta característica lo hace altamente legible y fácil de depurar. Sin embargo, esto tiene el inconveniente de hacer los mensajes más largos.

Los mensajes tienen la siguiente estructura:

- **Línea inicial** (termina con retorno de carro y un salto de línea)
  - Peticiones: Acción requerida, URL y versión HTTP
  - Respuestas: Versión HTTP, código de respuesta y una frase asociada al retorno de la respuesta.
- Las **cabeceras** del mensaje que terminan con una línea en blanco.
- **Cuerpo** del mensaje (es opcional)

Versiones del protocolo HTTP

- **HTTP/1.1** (junio de 1999) :Versión más usada actualmente. Las conexiones persistentes están activadas por defecto y funcionan bien con los proxies. También permite al cliente enviar múltiples peticiones a la vez por la misma conexión (pipelining).
- **HTTP/2** (mayo de 2015): Esta nueva mejora como se empaquetan los datos y en el transporte. Los exploradores más importantes solo soportan HTTP 2.0 sobre TLS.
- **HTTP/3** (Octubre de 2018): Es el sucesor propuesto de HTTP/2, que ya está en uso en la web, utilizando UDP en lugar de TCP para el protocolo de transporte subyacente. Al igual que el HTTP/2, no es obsoleto en las versiones principales anteriores del protocolo. El soporte para HTTP/3 fue agregado a Cloudflare y Google Chrome en septiembre de 2019, y puede ser habilitado en las versiones estables de Chrome y Firefox.

6.1. Estructura de Mensajes HTTP: Peticiones y Respuestas

En este punto nos centraremos en las peticiones y respuestas que constituyen la base de la comunicación en la World Wide Web. Analizaremos en detalle la sintaxis y los componentes esenciales de estos mensajes, desentrañando el protocolo HTTP para comprender cómo las solicitudes transmiten las intenciones del cliente y cómo las respuestas ofrecen los recursos correspondientes.

Mensajes de petición HTTP

Cuando el cliente realiza una petición (request), el mensaje incluye una cabecera que detalla el método o comando, junto con otros parámetros esenciales como la fecha, el código de transferencia, los lenguajes aceptados y el agente o programa cliente. Esta cabecera, estructurada de manera informativa, proporciona detalles cruciales para la comprensión y el procesamiento adecuado de la solicitud por parte del servidor.

En la actualidad, la versión más ampliamente adoptada de HTTP es la versión 1.1. Este estándar ha demostrado ser robusto y eficiente, sirviendo como la base para la mayoría de las interacciones web. Permite al cliente enviar múltiples peticiones a la vez por la misma conexión (pipelining) lo que hace posible eliminar el tiempo de Round-Trip delay por cada petición.

Los métodos más usados en una cabecera HTTP versión 1.1 son:

- **GET**: solicitud de un recurso especificado. Las solicitudes que usan GET solo deben recuperar datos y no deben tener ningún otro efecto.
- **HEAD**: Solicitud similar a una petición GET, pero recupera sólo la cabecera (o encabezado) de un recurso y no todo el contenido.
- **POST**: Envía datos para que sean procesados por el Servidor. Ejemplos: Los datos de un formulario, para subir un fichero, etc.
- **PUT**: Envía datos al servidor para que sean modificados. Mientras que POST está orientado a la creación de nuevos contenidos, PUT está más orientado a la actualización de los mismos (aunque también podría crearlos).
- **DELETE**: Envía una petición para borrar el recurso especificado.
- **OPTIONS**: Devuelve los métodos HTTP que el servidor soporta. Esto puede ser utilizado para comprobar la funcionalidad de un servidor web.
- **CONNECT**: Se utiliza para saber si tenemos acceso a un host bajo condiciones especiales, como por ejemplo: cifrados requeridos con SSL.

- **TRACE**: Este método se utiliza con fines de depuración y diagnóstico. El cliente puede ver lo que llega al servidor y de esta forma ver todo lo que añaden al mensaje los servidores intermedios
- **PATCH**: Su función es la misma que PUT, el cual sobrescribe completamente un recurso. Se utiliza para actualizar, de manera parcial una o varias partes.

Mensajes de respuesta HTTP

Una vez que el servidor ha procesado la solicitud recibida, se genera una respuesta (response) que típicamente está compuesta por dos partes fundamentales: **la cabecera** y **el cuerpo**.

En la cabecera de la respuesta, se especifican diversos detalles, incluyendo la versión del protocolo HTTP utilizada, **el código de respuesta** que indica el estado de la solicitud, el tipo de servidor que gestiona la comunicación, la fecha del archivo y otros parámetros pertinentes. Esta información detallada en la cabecera es fundamental para que el cliente comprenda y procese de manera adecuada la respuesta recibida del servidor.

El código de respuesta, representado por un número, se convierte en un elemento clave para interpretar el resultado de la petición. Este número proporciona una indicación rápida del estado de la solicitud, ya sea que se haya procesado exitosamente, haya encontrado algún tipo de error, o requiera una acción adicional.

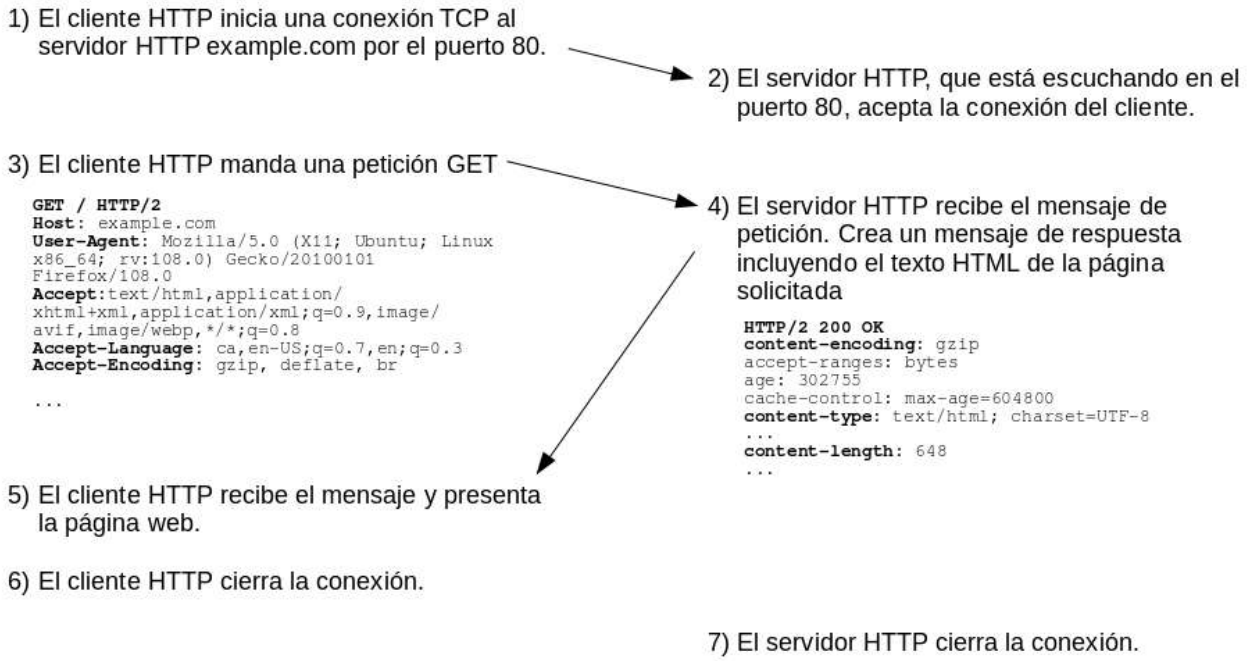
Códigos de respuesta:

- Códigos del **100** al **199**: Respuestas informativas. Indica que la petición ha sido recibida y se está procesando.
- Códigos del **200** al **299**: Respuestas correctas. Indica que la petición ha sido procesada correctamente.
- Códigos del **300** al **399**: Respuestas de redirección. Indica que el cliente necesita realizar más acciones para finalizar la petición.
- Códigos del **400** al **499**: Errores causados por el cliente. Indica que ha habido un error en el procesado de la petición a causa de que el cliente ha hecho algo mal.
- Ejemplo: **404** Not Found
- Códigos del **500** al **599**: Errores causados por el servidor. Indica que ha habido un error en el procesado de la petición a causa de un fallo en el servidor.

La cabecera y el código de respuesta trabajan en conjunto para ofrecer una comunicación precisa y detallada entre el cliente y el servidor en el entorno HTTP.

Ejemplo de comunicación HTTP

Protocolo HTTP para acceder desde un cliente a la página https://example.com/



HTTP sin Manejo de Estados

El protocolo HTTP opera sin manejo de estados, considerando cada petición como una transacción independiente sin relación con solicitudes anteriores.

Cada conexión es tratada como una transacción independiente que no tiene relación con cualquier solicitud anterior: el cliente hace una solicitud, el servidor responde y cierra la conexión. Para cada elemento que se transfiere (imágen, vídeo, etc.), se realiza una conexión independiente a través de sockets con destino al mismo puerto del servidor.

Las sesiones se guardan temporalmente y se pierden al cerrar el navegador o transcurrir un tiempo, aunque cierta información puede persistir en la parte del cliente mediante el uso de cookies.

Este enfoque define la naturaleza transaccional y efímera del protocolo HTTP.