2023
2024

Módulo: Despliegue de Aplicaciones Web

Unidad de Trabajo: 3

Implantación de arquitecturas web

Miguel Ángel García Blanes
IES Doñana

GOBIERNO DE ESPAÑA MINISTERIO DE EDUCACIÓN Y FORMACIÓN PROFESIONAL
UNIÓN EUROPEA
Fondo Social Europeo
El FSE invierte en tu futuro
Programa financiado por el Ministerio de Educación y Formación Profesional y cofinanciado por el Fondo Social Europeo

Contenido de la unidad

1 El concepto de web y su evolución ____ *3*

1.1 El concepto de web ____ 3

1.2 Evolución de la web ____ 4

2 Elementos de la arquitectura cliente-servidor ____ *5*

2.1 Arquitectura de dos niveles ____ 5

2.2 Arquitectura de tres niveles ____ 6

3 Aplicaciones web y aplicaciones de escritorio ____ *6*

3.1 Ventajas del software web ____ 7

3.2 Desventajas del software web ____ 7

4 Tecnologías usadas en aplicaciones web ____ *7*

4.1 En el lado servidor ____ 7

4.2 En el lado cliente ____ 8

4.3 En el lado cliente ____ 8

5 Servidores y aplicaciones libres y propietarias ____ *8*

6 Protocolo HTTP vs HTTPS ____ *9*

7 Servidores de aplicaciones ____ *9*

8 Instalación y configuración básica del servidor Apache ____ *9*

8.1 Definición y características ____ 9

8.2 Instalación y configuración del servidor Apache en Ubuntu ____ 10

8.3 Arranque y parada de Apache ____ 10

8.4 Estructura de Apache2 ____ 11

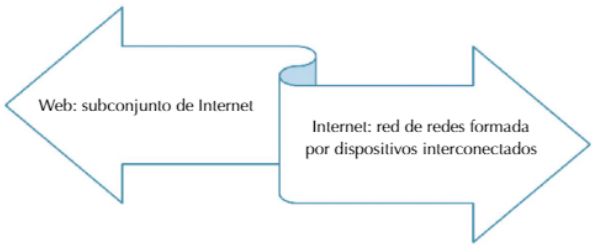
1 El concepto de web y su evolución

1.1 El concepto de web

La búsqueda del término web en Google arroja más de 400.000 resultados. Puede hacer referencia tanto al nombre común de la World Wide Web como a las páginas web. No se debe confundir la Web o World Wide Web con Internet.

Toma nota:
El World Wide Web Consortium (W3C) es una comunidad internacional que desarrolla los estándares que aseguran el crecimiento de la Web a largo plazo. En su página web puedes encontrar, entre otra información, estándares para el diseño de aplicaciones o web de los servicios o de los dispositivos. Puedes visitar su página web (www.w3c.es).

La Web, por tanto, es un subconjunto de Internet. En Internet existen multitud de servicios, como por ejemplo FTP (file transfer protocol) o el correo electrónico (e-mail).



La web convencional, que es visible para todos los usuarios, se conoce también con el término de **Surface Web**. Para poder navegar utilizaremos los browser o navegadores; mencionaremos Mozilla Firefox, Opera y Safari, entre otros, aunque sin lugar a duda Google Chrome es el más usado, tanto porque asegura una velocidad mayor, como por sus grandes funcionalidades.
La interfaz de Google Chrome presenta los siguientes elementos:

Barra de direcciones	Dirección página web, realizar búsquedas.
Pestañas	Para visitar varias páginas web.
Botones	Distintas funcionalidades desde actualizar la página hasta retroceder o avanzar en la navegación. También permiten la configuración y cambiar el aspecto del navegador.
Marcadores	Permiten realizar un acceso directo a una página.

Cuando se usan los navegadores, es frecuente confundir los términos ventana y pestaña. Las pestañas permiten cambiar rápidamente de tarea y abrir diferentes páginas web en una misma ventana. En el caso de Chrome es posible abrir todas las pestañas que se desee, alternar entre ellas, moverlas de lugar, personalizarlas o navegar de incógnito. Algunas combinaciones útiles para trabajar con este potente navegador se muestran en el siguiente cuadro

Combinación de tecla	Funcionalidad
CTRL + T	Pestaña nueva
CTRL + N	Ventana nueva
CTRL + MAYUS + N	Ventana nueva en modo incógnito
CTRL + (NÚMERO)	Visualizamos el contenido de esa pestaña. Por ejemplo, si se pulsa Ctrl y el número 3 visualizamos la tercera pestaña
CTRL + MAYUS + T	Permite recuperar pestañas cerradas
CTRL + W	Cerrar una pestaña

Todos los navegadores permiten su personalización instalando extensiones, temas y aplicaciones. En el caso de Chrome, para poder instalar una extensión hay que acceder a Chrome Web Store; una vez seleccionada la extensión, se instala de forma automática y, a partir de ese momento, es posible utilizarla junto con nuestro navegador. De la multitud de extensiones existentes, destacamos, por ejemplo: Adblock Plus para evitar anuncios indeseados.

Es importante recordar que no es lo mismo Internet que la World Wide Web. Dentro de Internet hay multitud de servicios como el correo electrónico, el servicio de transferencia de archivos o el servicio de videoconferencia.

1.2 Evolución de la web

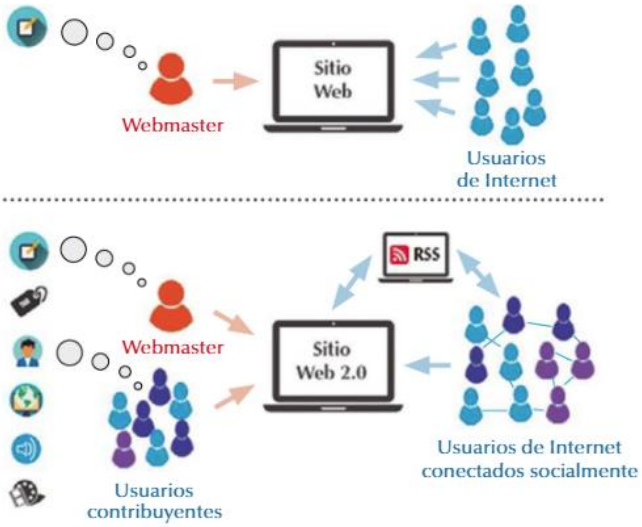
Cuando hablamos de la Web no sólo utilizamos ese término, sino que hablamos de Web 2.0, Web 3.0 e incluso Web 4.0. Ahora bien, ¿qué significan esos números?, ¿son versiones de la Web?

La evolución de la Web ha supuesto una transformación tanto en el modo de utilizarla como en las tecnologías que se han ido utilizando.

En etapas iniciales la Web se conocía con el nombre de **Web 1.0**. Tenía una finalidad principalmente divulgativa y las páginas eran estáticas, por lo que su contenido permanecía invariable a lo largo de sucesivas ejecuciones.

En el año 2004, Tim O'Reilly acuña el término **Web 2.0**, también conocida como web social, puesto que el usuario tiene un papel relevante, al crear y compartir información. El usuario ya no solo lee páginas, sino que interactúa con ellas. Aparecen las redes sociales, los blogs, las wikis, las páginas web..., que son fundamentalmente dinámicas e interactivas. En la Web 2.0 los usuarios han dejado de mirar cosas para empezar a hacer cosas en Internet.

No hay que concebir las nomenclaturas Web 1.0 o Web 2.0 como versiones de la web, sino como el reflejo de la evolución de la web. Es posible observar de manera gráfica los aspectos más característicos de la Web 1.0 frente a Web 2.0:



Todas las páginas web que se cataloguen como Web 2.0 reúnen los siguientes requisitos:

1. **Aplicaciones ricas en Internet** (RIA, Rich Internet Applications). Las aplicaciones web ofrecen prestaciones casi idénticas a las de escritorio. Ejemplo: Office 365
2. **Arquitectura orientada al servicio** (SOA, Service Oriented Architecture). En este tipo de arquitectura los componentes software son diseñados con el fin de usarse como un servicio en la red.
3. **Web social**. En las aplicaciones Web 2.0 el usuario es el centro de estas.

La Web ha ido incrementando su popularidad debido a las grandes prestaciones que nos ofrece. De forma continua los servicios van evolucionando y surgen nuevas tecnologías. Podemos decir que en la actualidad las tecnologías y los servicios existentes propician que sea nombrada con el término **Web 3.0** o Web semántica. El objetivo principal es acercar al usuario lo más posible al lenguaje natural y adaptar la navegación a las preferencias que este tiene. Con la Web 3.0 han surgido nuevas formas de búsqueda y de almacenamiento, y nuevos lenguajes, sobre todo de inteligencia artificial. Los cambios en nuestra forma de utilizar la Web y obtener resultados se perciben claramente mediante la publicidad selectiva que obtenemos en función de nuestras experiencias de navegación.

Por tanto, estamos ante una web más inteligente que analiza diferentes factores, como preferencias y hábitos de navegación. En solo una década ha habido un crecimiento exponencial tanto del número de sitios creados, como del número de usuarios que utilizan la web.

Ahora bien, la Web no ha parado de evolucionar. Actualmente vamos hacia lo que se denomina la **Web 4.0**. Con este nuevo modelo el usuario no solo interactuará con la web en búsqueda de información, sino que esta será capaz de mostrarle soluciones completas a sus necesidades.

Por ejemplo: "Quiero cenar sushi esta noche a las 9" y que automáticamente se envíe la petición al restaurante, sin necesidad de más intervención del usuario.

Por otro lado, la exponencial evolución de la Web y la cantidad masiva de datos generados en Internet han generado el auge de lo que se conoce con el nombre de Big Data. Podemos definir **Big Data** como el procesamiento masivo de datos de

diferente naturaleza que requieren una alta velocidad de proceso. Big Data suele ser definido a través de las llamadas 3V. Las 3V son: volumen (cantidad de datos), velocidad (procesamiento de datos) y variedad (tipos diferentes de datos).
En la actualidad, la utilidad del Big Data es incuestionable para todos los sectores, desde la banca, los seguros, la publicidad, la automoción e incluso la medicina. Teniendo en cuenta la evolución imparable de generación de información, se estima que en el 2025 alcanzaremos unos 163 zettabytes; el Big Data tendrá una importancia estratégica en todo tipo de sectores para poder gestionar tal cantidad masiva de información.
Ver documental: "Big Data: el valor de nuestra información"

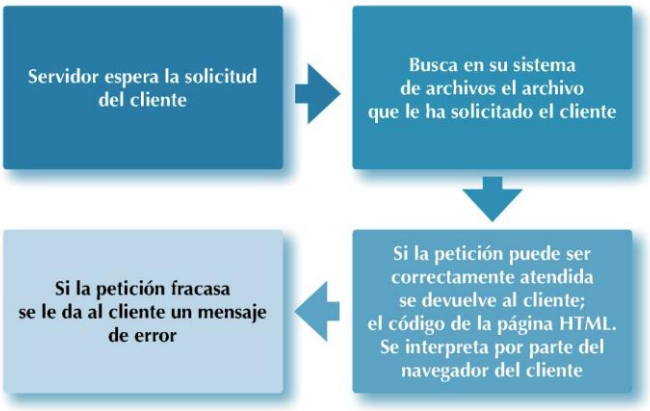
2 Elementos de la arquitectura cliente-servidor

Los elementos de este tipo de arquitectura varían en función de si estamos ante una arquitectura de dos capas o de tres. En cualquier caso, la arquitectura cliente-servidor es un modelo de aplicación distribuida con dos componentes principales: el servidor y el cliente. El servidor tiene como tarea principal proporcionar respuestas a las peticiones de los clientes. Hay muchos ejemplos de este tipo de arquitectura y de las cueles ya hemos visto algunas: DHCP, DNS, FTP.
En la siguiente tabla, se recogen algunas ventajas e inconvenientes de la arquitectura cliente / servidor:

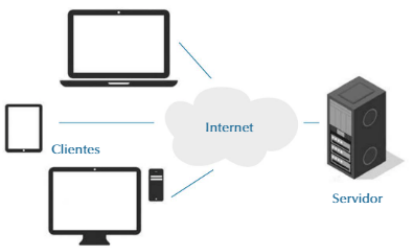
Ventajas	Desventajas
Escalabilidad, puesto que es posible aumentar la capacidad tanto de servidores como de clientes.	Posible congestión del tráfico en la Red, ya que existen gran cantidad de peticiones simultáneas al mismo servidor. Esto puede ocasionar un grave problema.
Disponibilidad, en el sentido de que, al estar distribuidas las funciones y las responsabilidades entre varios ordenadores independientes, es posible reemplazar, reparar, actualizar o incluso trasladar un servidor, sin que sus clientes se vean afectados por ese cambio.	Potentes recursos hardware en el servidor para que sea capaz de procesar de forma óptima las peticiones de los clientes.
Control centralizado por parte del servidor.	Mantenimiento complejo.

2.1 Arquitectura de dos niveles

Cuando hablamos de arquitectura de dos niveles estamos describiendo un sistema con dos elementos: cliente y servidor. En este modelo el servidor, polivalente, resuelve las peticiones que realizan los clientes. En el mundo web podemos entender este tipo de arquitectura secuenciando el proceso por el cual un usuario solicita una página web y la recibe en su ordenador en las etapas reflejadas en la siguiente imagen:



La siguiente imagen nos permite entender de forma gráfica cuál es la estructura lógica de este tipo de arquitectura.



Ver vídeo: Modelo Cliente Servidor, Explicación Simple (https://youtu.be/49zdlyLSwhQ)

Si hay una gran cantidad de procesos o son muy complejos, el modelo de dos capas no resulta eficaz. Se produce una gran congestión en la Red cuando los clientes tienen que acceder a descargar los datos del servidor. A este problema se le denomina cliente pesado. La arquitectura de tres niveles intenta evitar la sobrecarga equilibrando las tareas que tiene que soportar el servidor.

2.2 Arquitectura de tres niveles

Para conseguir soluciones más seguras, flexibles y escalables es necesario optimizar el uso de recursos. Para ello hay que repartir las funciones en tres niveles:

1. **Cliente.** Es el ordenador que, a través de la interfaz de usuario, realiza la petición de recursos al servidor.
2. **Servidor de aplicaciones.** El servidor hará uso de otro servidor para poder resolver las peticiones de recursos realizadas por los clientes.
3. **Servidor de datos.** Mediante el nivel de la base de datos es posible darle los datos al servidor de aplicaciones. Existen múltiples interfaces entre los dos servidores, dependiendo de los lenguajes de programación y del tipo de base de datos.

A medida que crecen las aplicaciones web, mantenerlas resulta más difícil. El paradigma modelo-vista-controlador (MVC) separa en diferentes niveles o capas la arquitectura de una aplicación software. En el caso de las aplicaciones web, las capas estarían formadas por: la vista (o sea la página en HTML), el controlador (el programa que produce el contenido HTML y que obtiene de forma dinámica los datos) y el modelo (en esta capa se trabaja con los datos, por lo que normalmente habrá una base de datos que contiene diversa información almacenada). Algunas de las ventajas de MVC son: la separación de los datos de su representación visual, la escalabilidad y el control de errores.

Ver más información sobre la arquitectura del software multicapa: https://youtu.be/kHvxX1E9vIU

3 Aplicaciones web y aplicaciones de escritorio

Aquellas aplicaciones que se desarrollan en un sistema operativo específico, para ser ejecutados bajo este, son conocidas como **aplicaciones de escritorio**. Estas aplicaciones serán más o menos eficaces en función de la configuración hardware del ordenador donde se instalen. Ejemplos de aplicaciones de escritorio son: Word, Notepad, Paint, entre otras.

Las aplicaciones de escritorio son las que tradicionalmente tenemos instaladas en nuestros ordenadores personales o de empresa. Existen aplicaciones tanto de pago como libres. Algunos de los problemas que pueden producirse al usarlas son:

1. Se puede producir incompatibilidad entre versiones.
2. Puede resultar difícil la instalación y la actualización.
3. Su coste es elevado.
4. En el caso de un ámbito empresarial, podemos encontrarnos con ordenadores que tengan diferentes versiones de un programa. Como consecuencia puede haber tanto pérdida de funcionalidad como problemas de consistencia.
5. Es posible que haya falta de portabilidad de la aplicación a diferentes sistemas operativos.

Ahora bien, las aplicaciones de escritorio son ampliamente utilizadas en todos los ámbitos por su gran cantidad de prestaciones y por su elevado tiempo de respuesta.

Definiremos **aplicación web** como toda aquella aplicación accesible a través de un navegador. Existen multitud de ejemplos de aplicaciones web: la Wikipedia, el correo electrónico o un servicio de comercio electrónico.

3.1 Ventajas del software web

Las aplicaciones Web han ido avanzando en cuota de mercado tanto por sus prestaciones como por sus ventajas. Algunas de las ventajas más destacables:

1. **Ausencia de costes de actualización**. Como son los navegadores los que visualizan las páginas web es suficiente con realizar las actualizaciones en el servidor.
2. **Datos centralizados**. El servidor, no solamente aloja las páginas web, sino que también aloja los datos. Estos últimos suelen estar almacenados en una base de datos centralizada.
3. **Ausencia de instalación**. No es necesario instalar nada para usar una aplicación web, solo es necesario un navegador para acceder al servicio.
4. **Actualización constante**. Los equipos siempre acceden a la última versión actualizada de la aplicación.
5. **Movilidad**. Acceso desde cualquier ubicación con conexión a Internet.

Toma nota:
Actualmente las aplicaciones web tienen prácticamente las mismas funcionalidades que las de escritorio.

3.2 Desventajas del software web

Las ventajas anteriormente expuestas pueden hacernos pensar que las aplicaciones de escritorio son claramente inferiores a las de la Web. Antes de decantarnos por un tipo u otro debemos conocer las siguientes desventajas de las aplicaciones Web:

1. **Menor potencia**. Normalmente las aplicaciones de escritorio suelen tener mejores prestaciones y ser menos potentes. Por ejemplo, podemos pensar en Office de escritorio y Google Docs.
2. **Infrautilización del hardware**. Las aplicaciones web no supeditan su rendimiento a unas mejores prestaciones de hardware del ordenador.
3. **Conectividad rápida y constante**. Para utilizar todas las prestaciones del software web es necesario un acceso rápido y fiable a Internet. En la actualidad, esto no es muy preocupante, ya que está ampliamente extendida la conectividad a Internet, y por otra parte muchas aplicaciones web pueden ser ya utilizadas en modo offline.

4 Tecnologías usadas en aplicaciones web

Actualmente, la mayoría de las aplicaciones web del mercado usan páginas dinámicas, que se ejecutan en el servidor web y se visualizan en el cliente (navegador). Como se comentó anteriormente, existen páginas de contenido estático y dinámico; normalmente, cuando existe contenido dinámico se ejecuta código tanto en el cliente como en el servidor. Se va a diferenciar el código que se ejecuta en estos dos nodos:

4.1 En el lado servidor

- **CGI (Common Gateway Interface)**: al principio de Internet, el servidor solamente podía ejecutar programas del tipo C, Perl y líneas de comando Powershell. Estas instrucciones eran ejecutadas por el sistema operativo y se transmitían al navegador mediante el CGI.
- **ASP.NET** (Active Server Pages): es un framework de desarrollo libre de Windows orientado a objetos. Aunque existen versiones para Linux y Unix.
- **Java**: existe un gran grupo de tecnologías desarrollado por Sun Microsystems que se pueden ejecutar en el servidor, y algunas son JSF (JavaServer Faces), JSP (JavaServer Pages) y los servlets. De estas tres, la que más se usa es JSP, ya que es la más extendida y es compatible con casi todos los servidores web.
- **Ruby**: es un lenguaje interpretado de propósito general, también es dinámico y flexible. Además, es de alto nivel, y lo más importante es que es software libre y multiplataforma.
- **Perl**: generalmente este lenguaje de programación se ejecuta en el servidor. Apache, por ejemplo, tiene un módulo que permite ejecutar programas de este tipo. Toma muchas características del lenguaje C y se caracteriza por su destreza a la hora de procesar texto.
- **PHP**: es un lenguaje de propósito general del desarrollo backend más usado y popular que existe en el mercado. Es útil en el desarrollo de aplicaciones y motor de Wordpress, Drupal, Magento, Joomla, etc. Además, existen frameworks muy potentes como Laravel o Symfony. Por último, una de las mejores ventajas es que accede fácilmente a bases de datos como Oracle o Mysql, y también a bases de datos NoSQL como MongoDB.
  PHP tiene un inconveniente, ya que en algunas funcionalidades tiene agujeros de seguridad si no se toman las medidas oportunas.

- **Python**: es un lenguaje interpretado muy poderoso, fácil de aprender y de alto nivel. Se usa en multitud de aplicaciones y sobre todo en aplicaciones de seguridad. Es un lenguaje orientado a objetos, que destaca por su tipado dinámico.
- **JavaScript**: es un lenguaje ligero, interpretado y orientado a objetos. Para aprenderlo es preciso un conocimiento básico de HTML y CSS. Interviene sobre todo en el contenido dinámico y en la interacción con el usuario. Permite controlar archivos de multimedia, creación de imágenes animadas, animación en 3D. Destaca por su aportación a la mayoría de los ámbitos de la tecnología.

4.2 En el lado cliente

- **HTML y CSS**: HTML es el lenguaje de marcas que se inventó junto con el navegador y el medio para transmitir información entre el cliente y el servidor. Posteriormente, surgió CSS (Cascading Style Sheets) que permitirá diseñar gráficamente la página.
  HTML como tal, posee una especificación que parte del W3C. Comenzó en la versión 1 y ya se ha publicado la versión 5.
- **JavaScript**: comentado anteriormente, se puede usar tanto en el cliente como en el servidor. Es uno de los más extendidos porque es soportado por la mayoría de los navegadores.

4.3 En el lado cliente

- **XML:** es una tecnología relacionada con lenguajes de marca, y que permite compartir datos, incluso formar parte de una base de datos. Estos ficheros .xml se centran en la configuración de la mayoría de las aplicaciones y de los servidores web y de aplicaciones.

5 Servidores y aplicaciones libres y propietarias

En la actualidad, existe una serie de plataformas y servidores libres para poder programar sobre cualquier lenguaje de los vistos anteriormente. Por otro lado, en menor medida, pero también disponibles, están los servidores propietarios en los que se puede programar cualquier aplicación con base en cualquier lenguaje.
En esta sección se van a nombrar las más representativas plataformas en las que los desarrolladores programan sus aplicaciones. Para ello, se van a definir los componentes que forman la arquitectura web:

a) **Sistema operativo**: es fundamental porque es el software base sobre el cual se sustentan los demás componentes. Los principales son Windows, Linux, Unix y Mac.
b) **Servidor web**: es el servicio que va a atender las peticiones de los clientes y les va a responder lo más pronto posible.
c) **Bases de datos**: es la principal fuente de información de la que se nutre el servidor web para mostrar al cliente la información que solicita. Este componente es básico en cualquier portal de hoy en día. Existen dos grandes grupos, SQL y NoSQL.
d) **Lenguaje de programación:** es el lenguaje que se encarga de la lógica de la aplicación. Tiene como función recibir las peticiones de los clientes, consultar la información en la base de datos y responder a las peticiones. Dependiendo de la aplicación, se pueden usar varios lenguajes de programación en la aplicación delimitando sus funciones. La clave de que funcione todo correctamente es el encapsulamiento y la modulación de los componentes de la aplicación.

Teniendo en cuenta los cuatro componentes anteriores, existen una plataforma clave para trabajar en el desarrollo de una aplicación. Se denomina XAMPP (https://www.apachefriends.org), y es conocidas por su popularidad y su uso en gran parte del entorno productivo empresarial.
La plataforma XAMPP es de software libre e incluye los cuatro componentes descritos anteriormente, de ahí su popularidad y su crecimiento exponencial en el desarrollo de aplicaciones.
Sus iniciales provienen de los siguientes componentes:

- X: cualquier sistema operativo
- Apache: servidor web.
- MariaDB: base de datos. En versiones más antiguas incluía MySQL
- PHP, Perl: lenguaje de programación del servidor.

6 Protocolo HTTP vs HTTPS

Como se ha comentado anteriormente, el protocolo por antonomasia es HTTP y el seguro es HTTPS. En este apartado se entrará en profundidad en cada uno de ellos.

HTTP es un protocolo de la capa de aplicación que **escucha por el puerto 80** basado en el modelo cliente-servidor. Se basa en TCP, por lo tanto, es orientado a conexión y escucha las peticiones de los clientes. Una vez aceptada la conexión, se encarga de mantenerla y garantizar la transferencia de datos sin errores.

Por otro lado, **HTTPS** es un protocolo de la capa de aplicación que **escucha por el puerto 443**, y también se basa en el modelo cliente-servidor, pero en modo seguro. Se basa en TCP, es orientado a conexión y encripta los datos para asegurar una transmisión segura entre los dos extremos. Para realizar esta encriptación necesita de certificados, siempre y cuando sean válidos. Sí es cierto que al cifrar la información en el servidor y descifrarla en el cliente se pierde tiempo de computación. Aunque solamente se necesite cifrar cierta información.

Cualquier servidor web medio serio, por ejemplo, Apache, posee la acción de emitir certificados. En este entorno, la parte fundamental son los navegadores, ya que deben validar dichos certificados a partir de entidades certificadoras. Es necesario tener una entidad que sea fiable, por ejemplo, la FNMT que es la encargada en España de suministrar los certificados a los ciudadanos.

El protocolo HTTPS usa la encriptación SSL (Secure Socket Layer) que encripta datos sensibles (actualmente se le llama TLS). Un certificado SSL o TLS se instala en el servidor.

7 Servidores de aplicaciones

En el mercado actual tan cambiante, existen multitud de servidores preparados para ser instalados en cualquier plataforma. A continuación, se mostrarán algunos de los servidores web más usados, que permiten ejecutar en su entorno diferentes tecnologías:

- **Apache:** es uno de los servidores más usados en Internet, su nombre completo es Apache HTTP Server. Su primera versión fue por 1996, desde entonces ha sido un referente como servidor, pero tiene dos grandes competidores que son Nginx y Microsoft IIS.
  Tiene como ventaja que su desarrollo es código abierto, y es multiplataforma. Entre sus desventajas se encuentra su bajo rendimiento cuando se realizan multitud de peticiones por parte de los clientes.
- **Nginx**: es un servidor de código abierto y gratuito que nació en 2004, y es uno de los más usados por las compañías en el mundo. Destaca por su alto rendimiento e incluye funciones como balanceador de carga, POP3 e IMAP.
  Se puede instalar en Windows, Linux y Unix. Es altamente escalable, por si fuera necesaria una ampliación, y consume muy pocos recursos para resolver las peticiones de los clientes.
- **Microsoft IIS**: es el software de Microsoft conocido por Internet Information Services. Comenzó en Windows NT. Permite el procesamiento de páginas del tipo ASP, ASP.net, PHP o Perl. Está programado en C++, pero tiene una gran desventaja y es que solamente se puede instalar en Windows, con lo que conlleva una amenaza de seguridad constante. Además, posee servicios como FTP y SMTP.

8 Instalación y configuración básica del servidor Apache

Como hemos comentado anteriormente, Apache es uno de los servidores de aplicaciones web más usados por las grandes compañías. Por ello vamos a tratarlo con más de profundidad y vamos a instalarlo sobre una MV con el sistema operativo Linux

Posteriormente veremos algunos datos básicos para su configuración inicial

8.1 Definición y características

Apache HTTP Server es un software gratuito y de código abierto para plataformas Linux o Unix y Windows. Es desarrollado y mantenido por Apache Software Foundation. Algunas de las características que nos ofrece este servidor serían:

- Autenticación y autorización.
- Host virtuales.
- Modularización.
- Seguridad mediante cifrado.
- Monitorización mediante archivos de logs.
- Páginas web estáticas y dinámicas.

8.2 Instalación y configuración del servidor Apache en Ubuntu

Lo más lógico sería instalar Apache en un sistema operativo de tipo servidor (Ubuntu Server) pero puede ser instalado en una versión estándar con interfaz gráfica.

Nosotros vamos a realizar una instalación manual para tener mayor control sobre el proceso y la configuración. Apache puede funcionar de forma autónoma, pero en la actualidad (más aun hablando de aplicaciones web) suele necesitar una base de datos y cada vez son más los sistemas que usan PHP. Por ello han proliferado los instaladores AMP (Apache, MySQL y PHP) que instalan y configuran los tres sistemas para que funcionen de manera conjunta. Esto requiere una gran configuración de los tres elementos para poder tener un servidor seguro. Para seguridad sobre PHP y MySQL habrá que referirse a los módulos propios.

XAMPP es la versión más usada para Windows y Linux. Dependiendo del sistema para el que vayan orientados, estos paquetes se llaman WAMP y LAMP genéricamente. La instalación de estos paquetes es tremendamente sencilla.

Para instalar el servidor Apache en sistemas Linux Debian/Ubuntu, debes instalar el paquete apache2, ejecutando:

```
#apt update
#apt dist-upgrade
#apt install apache2
```

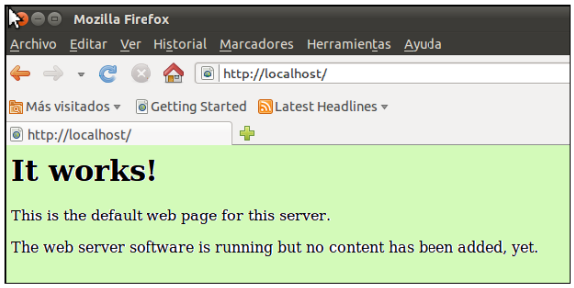
También puedes instalar el servidor apache mediante el gestor de paquetes Synaptic seleccionando e instalando el paquete apache2. Se instalarán o actualizarán otros paquetes de los que depende apache2 como apache2.x-common y apache2.x-bin.

Tras haber instalado el servidor, puedes probar su funcionamiento ya que durante la instalación se ha creado un documento llamado index.html y está almacenado en la carpeta /var/www que se ha configurado como carpeta raíz del sitio web. El servidor queda iniciado tras la instalación (si no ha ocurrido nada anormal), y ha sido configurado para que el documento index.html sea la página de inicio del sitio web.

Para comprobar que el servidor funciona, debes iniciar el navegador web en la misma máquina donde has instalado el servidor y dirígete a la URL:

http://localhost o http://127.0.0.1

También puedes hacer la prueba ejecutando el navegador en otra máquina de la red y dirigiéndote a la URL http://nombre_DNS_servidor o bien http://dirección_IP_servidor. En cualquiera de los casos, debes ver en el navegador una página como ésta:



8.3 Arranque y parada de Apache

Cuando se ha instalado el servidor Apache, éste queda iniciado y preparado para iniciarse automáticamente cada vez que se inicie el equipo servidor. En cualquier momento puedes detener, iniciar o reiniciar el servicio Apache para aplicar cambios realizados, por un funcionamiento anómalo o por cualquier otro motivo. En particular, cuando hayamos modificado los archivos de configuración de Apache debemos reiniciar el servicio.

Podemos controlar el servicio de la siguiente forma:

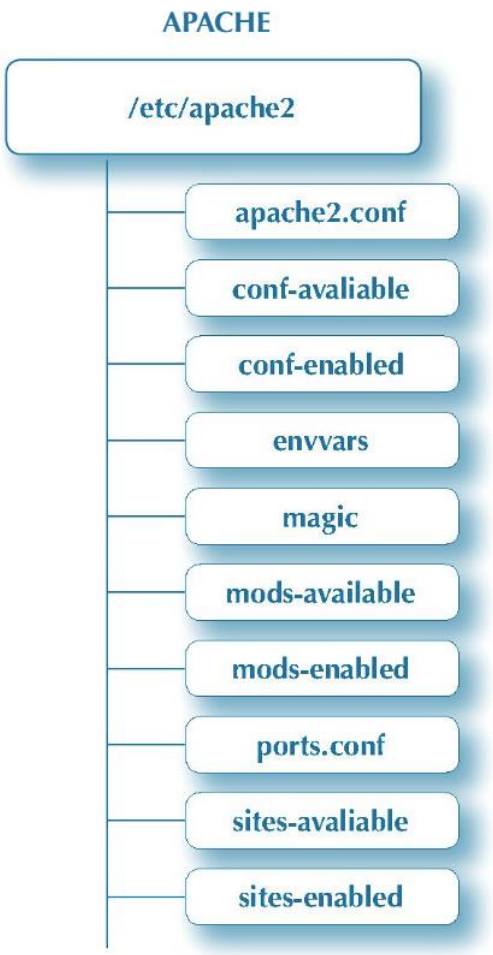
Acción	Comando
Comprobar el estado del servicio.	`service apache2 status`
Detener el servicio.	`service apache2 stop`
Iniciar el servicio.	`service apache2 start`
Reiniciar el servicio.	`service apache2 restart`

Otros comandos útiles:

- **acachectl start**: inicia el servicio de apache2.
- **apachectl stop**: para el servicio de apache2.
- **apachectl restart**: reinicia el servicio apache2.
- **apachectl status**: muestra el estado del servicio apache2.
- **apachectl configtest**: ejecuta una prueba sintáctica para comprobar que los ficheros de configuración son correctos.

8.4 Estructura de Apache2

A continuación, se va a analizar la estructura que tiene apache2 cuando se instala, que es la que se puede ver en la figura adjunta. Se explicarán cada uno de los directorios y ficheros de configuración que posee apache2 y se realizarán las configuraciones básicas para que se observe cómo trabaja este servidor. Además, se darán unas nociones básicas de seguridad para que minimicemos la amenaza desde el exterior. Se comenta la estructura de la forma siguiente:



- **apache2.conf:** es el fichero de configuración principal del servidor apache2. Condene las variables globales. Cualquier cambio en este fichero implicaría reiniciar el servicio.
- **conf-available**: este directorio contiene configuraciones adicionales que están asociadas a un módulo en particular. Las configuraciones no están activas.
- **conf-enabled**: este directorio contiene enlaces al directorio anterior para poder activar las configuraciones que contiene.
- **envars:**: es un fichero en el que se definen variables de entorno como APACHE_RUN_USER, APACHE_RUN_GROUP, etc. que en principio no es necesario modificar.
- **mod-available, mod-enabled:** estos directorios son similares a los sites, pero para módulos que se pueden acoplar al servidor web.
- **ports.conf:** este fichero siempre está incluido en el fichero apache2.conf y permite configurar los puertos y las IP por las que escucha el servidor. El puerto por defecto es el 80.
- **sites-available:** este directorio contiene los ficheros de los hosts virtuales definidos en el servidor, que pueden ser diferentes. Estos sitios están disponibles, pero no activos.
- **sites-enabled**: este directorio es similar al anterior pero las definiciones de hosts virtuales que se están usando, normalmente son enlaces simbólicos a los ficheros que se encuentran en sites-available. Por defecto, tiene el directorio /var/www/html, que es donde se almacena la página principal de Apache.

Nota: cuando instalamos Apache y accedemos a http://IP-SERVER, intervienen los ficheros de configuración 000-default.conf y apache2.conf. La configuración de ambos es lo que permite el acceso de un usuario

000-default.conf es un sitio virtual que se configura en el servidor web. Dicho fichero contienen varias directivas, siendo una de las más importante **DocumentRoot**. Esta directiva permite configurar una ruta donde se ubicará el fichero index.html. Para ello se ha creado un host virtual que escucha por el puerto 80 con la directiva <VirtualHost *:80>.

```
<VirtualHost *:80>

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html


        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>
```

Para habilitar el sitio es necesario usar el comando **a2ensite** y para deshabilitarlo es el comando **a2dissite**. Aunque los hosts virtuales se tratarán en el próximo tema, es necesario comentarlo aquí para la configuración básica de Apache.
Las otras directivas relacionadas con esta configuración se encuentran en el fichero apache2.conf.

El primer bloque (<Directory />) deniega todos los accesos. Esto se realiza como método de seguridad para que nadie acceda al sistema de ficheros del sistema operativo. Dicho bloque, por defecto, tiene las directivas:

- **Options FollowSymLinks**: permite los enlaces simbólicos a otros directorios.
- **AllowOverride None**: ignora el fichero .htaccess para que no existan demasiadas llamadas a este fichero.
- **Require all denied:** deniega el permiso a todo el sistema de ficheros.

```
<Directory />
        Options FollowSymLinks
        AllowOverride None
        Require all denied
</Directory>

<Directory /usr/share>
        AllowOverride None
        Require all granted
</Directory>

<Directory /var/www/>
        Options Indexes FollowSymLinks
        AllowOverride None
        Require all granted
</Directory>
```

Y en el bloque <Directory /var/www>, se incluyen las directivas que afectarán al directorio /var/www:

- **Optiotons Indexes FollowSymLiuks:** Indexes permite la visualización del directorio en caso de que no exista en el directorio el fichero index.html o los patrones declarados en el servidor Apache. FollowSymLinks que permite los enlaces simbólicos a otros directorios.
- **AllowOverride None:** ignora el fichero .htaccess para que no existan demasiadas llamadas a este.
- **Require all granted**: da permiso a todo el mundo y desde cualquier IP a directorio /var/www.