MÓDULO DESPLIEGUE DE APLICACIONES WEB

TEMA 2

Servidores Web

CICLO DE GRADO SUPERIOR INFÓRMATICA

DESARROLLO DE APLICACIONES WEB

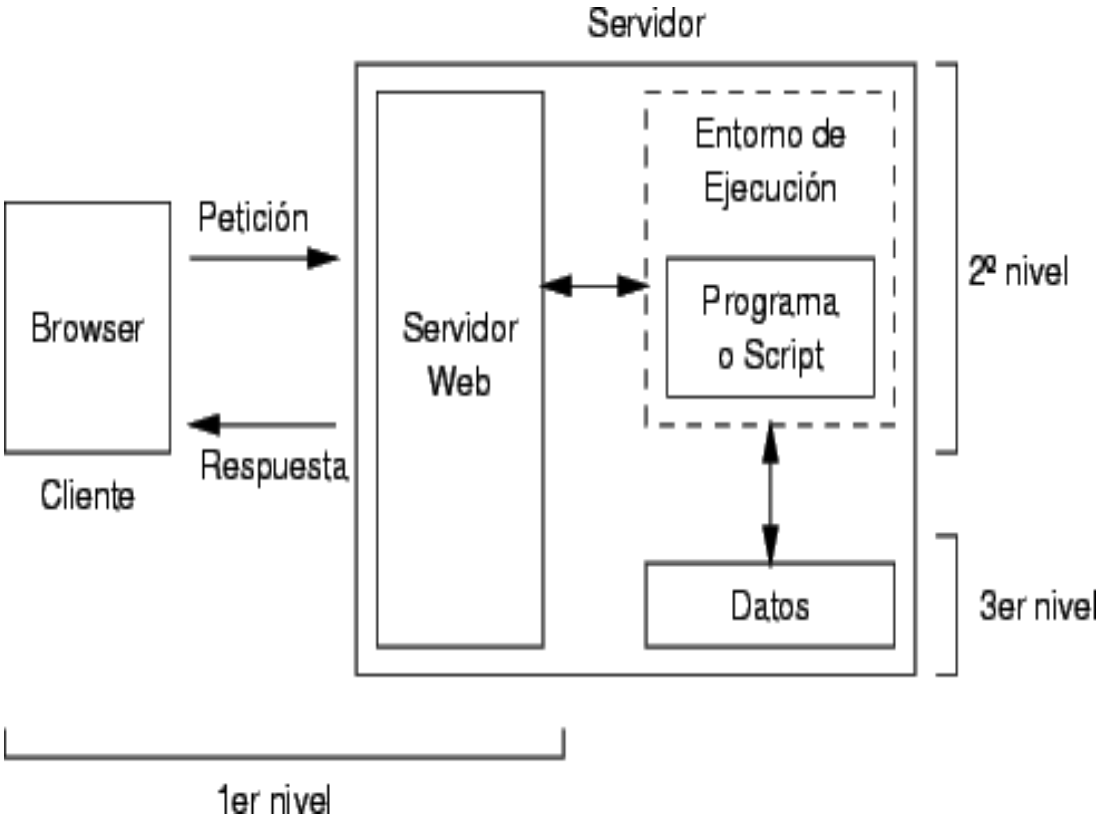
Índice

1. Introducción

En la anterior unidad se ha implementado un servidor web de diferentes maneras. Se han observado diferentes características de estos sistemas completos, pero se ha comentado poco sobre la función de PHP en ellos. Nos podríamos preguntar, ¿cómo funciona una página web dinámica con PHP?

podemos Se pude resumir que las páginas web dinámicas funcionan de la siguiente manera:

- El usuario se conecta a la web.
- El servidor web Apache recibe la solicitud del cliente (navegador).
    - Si solicita una página estática HTML, la busca y se la devuelve al cliente. Pero si debe ejecutar un script PHP, entonces:
        - Apache, carga PHP para ejecutarlo.
        - Si PHP necesita realizar alguna operación en una base de datos, entonces interacciona con el gestor (MySQLServer), enviándole las consultas y recuperando los datos.
        - PHP devuelve la respuesta a Apache.
- Apache devuelve la solicitud al cliente.



Ya sé cuál es el proceso y ahora, ¿cómo lo implemento? En realidad, el servidor web no es más que un programa que está en ejecución de forma continua en un ordenador, manteniéndose a la espera de peticiones de los clientes (navegadores web).

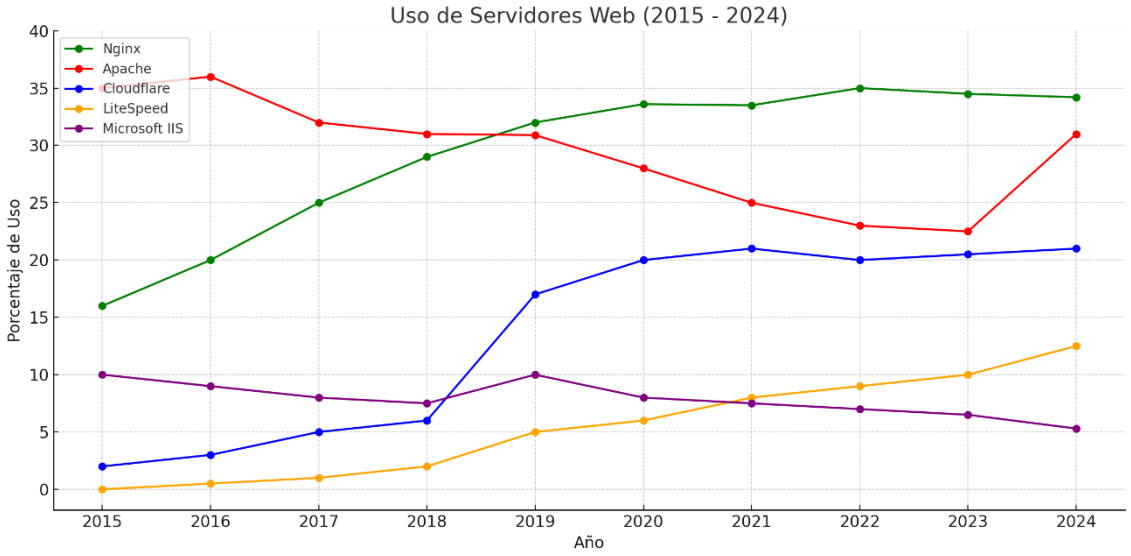
Pero si tenemos páginas web dinámicas, sabemos que entran en funcionamiento lenguajes de programación o interpretados (PHP) e incluso gestores de bases de datos (MySQL). Necesitaremos entonces:

- Servidor web Apache.
- Intérprete PHP.
- Sistema Gestor de Bases de Datos MySQL.

¿Por qué escoger Apache-MySQL y PHP?

Apache

Los servidores web más utilizados actualmente son **Nginx** y **Apache HTTP Server**. Ambos compiten por el liderazgo en el mercado de servidores web, aunque cada uno tiene características que los hacen más adecuados para diferentes escenarios.



Aunque Nginx lo utilizaremos en próximas prácticas, en un inicio se ha utilizado Apache ya que ha ganado una gran popularidad en los últimos años, y cuenta con las siguientes propiedades:

- Ha sido históricamente el servidor web más utilizado y sigue siendo una opción robusta, especialmente para proyectos que requieren gran flexibilidad.
- Es conocido por su configurabilidad y su capacidad para integrar una amplia gama de módulos.

- Suele ser la opción preferida en muchas distribuciones de servidores Linux y en entornos donde la compatibilidad con scripts y configuraciones es prioritaria.
- Implementa el protocolo HTTP, HTTPS e incorpora sitios virtuales.

Apache comenzó su desarrollo en 1995 basándose en NCSA y debe su nombre, por un lado, a que Behelendorf quería un nombre que simbolizase algo enérgico y firme y pensó en la tribu Apache. Por otra parte, inicialmente se trataba de un conjunto de parches para el servidor de NCSA (patchy server) que suena parecido a Apache.

El servidor se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation. Si esquematizamos y unimos alguna característica más a las comentadas anteriormente, podríamos diferenciar:

- Altamente configurable.
- Admite bases de datos con autenticación.
- Multiplataforma.
- Modular.
- Extensible.
- Dispone de interfaz gráfica.

En cuanto a su arquitectura podemos destacar lo siguiente:

- Estructurado en módulos.
- Cada módulo contiene un conjunto de funciones relativas a un aspecto concreto del servidor.
- El archivo binario httpd contiene un conjunto de módulos que han sido compilados.
- La funcionalidad de estos módulos puede ser activada o desactivada al arrancar el servidor.

Los módulos de Apache se pueden clasificar en tres categorías:

- **Módulos base**: Se encargan de las funciones básicas.
- **Módulos multiproceso**: Encargados de la unión de los puertos de la máquina, aceptando las peticiones y atendiéndolas.
- **Módulos adicionales**: se encargan de añadir funcionalidad al servidor.

PHP

Utilizar PHP para el diseño de una web dinámica tiene varias ventajas y razones que lo hacen una opción sólida para muchos proyectos, especialmente aquellos que buscan facilidad de uso, flexibilidad y una comunidad de soporte amplia. Algunas de las principales razones por las que elegir PHP para el desarrollo de sitios web dinámicos podrían ser las siguientes:

- Facilidad de Aprendizaje y Uso
- Amplia Disponibilidad de Hosting
- Desarrollo Rápido con Frameworks Populares
- Integración con Bases de Datos Relacionales
- Soporte para Plantillas Dinámicas y Generación de HTML
- Ampliación de Funcionalidades mediante Librerías
- Gran Comunidad y Soporte Extenso
- Seguridad y Mantenimiento
- Costo y Licencia de Código Abierto
- Compatibilidad con CMS Populares

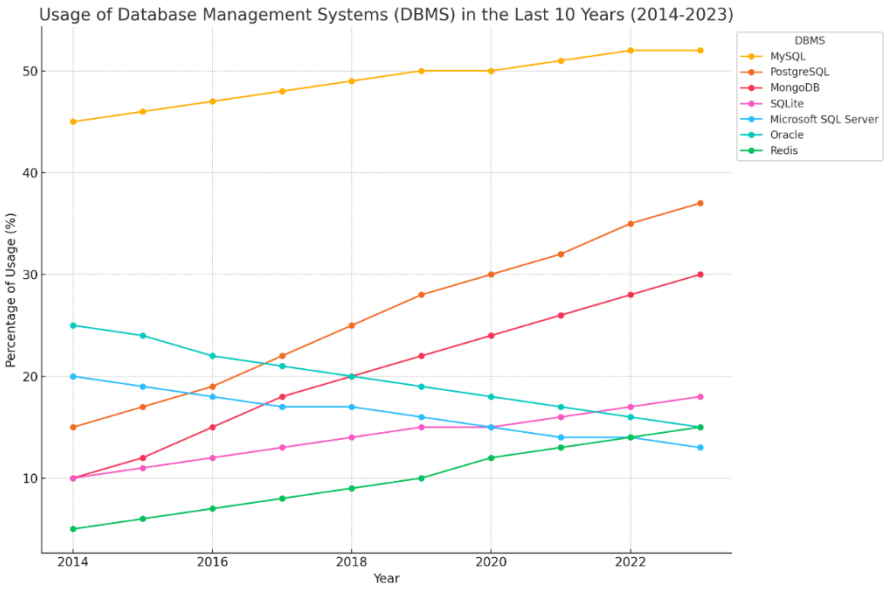
En conclusión, PHP sigue siendo una opción muy relevante para el desarrollo de sitios web dinámicos debido a su flexibilidad, la amplia disponibilidad de recursos y su capacidad para integrar funciones avanzadas de manera sencilla. Aunque han surgido otros lenguajes y tecnologías para la web, PHP se mantiene fuerte en el ecosistema gracias a su madurez, versatilidad, y su capacidad para adaptarse a las necesidades de los desarrolladores y empresas.

MySQL

Es un sistema gestor de bases de datos relacional, multihilo y multiusuario muy popular y usado.

Desde 2009 Oracle desarrolla MySQL como software libre, se ofrece como GNU GPL para cualquier uso que sea compatible con esa licencia. Pero si queremos incorporarlo a productos sin GNU GPL, entonces debemos compra la licencia.

En la siguiente gráfica se observa como MySQL es el líder destacado dentro de los sistemas gestores de bases de datos.



2. Arquitectura LAMP / WAMP /MAMP.

El Servidor **LAMP**, nos permitirá instalar aplicaciones web accesibles desde nuestra red local, y si abrimos el puerto 80 de nuestro router, también serán accesibles desde Internet. La gran mayoría de las aplicaciones web libres existentes, requieren de Apache + MySQL + PHP para funcionar. Podemos instalar estas aplicaciones por separado y después configurarlas, pero instalando un paquete LAMP se instalan y configuran automáticamente dichas aplicaciones.

La arquitectura **LAMP** es una plataforma de software libre para:

- Sistema Operativo **L**inux.
- **A**pache.
- Gestor **M**ySQL.
- Intérprete de **P**HP.

La versión LAMP para Microsoft se denomina arquitectura **WAMP**.

- Sistema Operativo Microsoft (**W**indows).
- **A**pache.
- Gestor **M**ySQL.
- Intérprete de **P**HP.

La versión LAMP para Mac OS X se denomina arquitectura **MAMP**.

- Sistema Operativo **M**ac OS X.
- **A**pache.
- Gestor **M**ySQL.
- Intérprete de **P**HP.

Todavía podemos simplificar más, incluyendo el módulo adicional PHPMyAdmin; se trata de una interfaz gráfica que facilita las tareas de administración del gestor MySQL cuando tenemos instalado el intérprete de PHP.

El paquete XAMPP es otra alternativa válida para los tres sistemas operativos anteriores, que incluye módulos opcionales: PHPMyAdmin, Filezilla FTP, Mercury Mail.

WAMP SERVER	Wamp Server utiliza las siguientes herramientas: Windows como Sistema Operativo, Apache, como servidor web, MySQL, como gestor de bases de datos y PHP como intérprete del lenguaje de programación PHP.	
LAMP SERVER	Lamp Server es un sistema análogo pero que funciona bajo el Sistema Operativo Linux.	
MAMP SERVER	Mamp Server es un sistema análogo pero que funciona bajo el Sistema Operativo Macintosh.	
XAMPP SERVER	Este paquete tiene la característica principal que se puede instalar para cualquiera de los diferentes sistemas opera-tivos, es decir, existen versiones tanto para Windows como para Linux, Macinstosh. Siendo, también, un sistema análogo al Wamp Server.	XAMPP

Entornos de trabajo

Cuando necesitamos escribir código PHP y crear páginas dinámicas, podemos plantear nuestro trabajo varias maneras:

- Trabajar en modo local o en red local, con algún paquete LAMP.
- Utilizar un servidor remoto, con alojamiento gratuito y con un cliente FTP para subir nuestras páginas.

Normalmente los paquetes LAMP se utilizan en los siguientes entornos de trabajo:

- Entornos de desarrollo.
- Entornos de integración.
- Entornos de preproducción.
- Entornos de pruebas.

Aplicaciones comunes

Apache + MySQL + PHP son la base para poder instalar infinidad de aplicaciones web libres, entre las que destacamos:

- **Gestores de Contenidos orientados a sitios web**: Joomla, WordPress y Drupal hasta la versión 7 (actualmente symphony)
- **Gestores de Contenidos orientados a educación**: Claroline y Moodle.
- **Wikis**: Mediawiki, Tikiwiki, Dokuwiki,...
- **Foros**: phpBB, myBB...
- **Galerías de imágenes**: Gallery, Coppermine

3. Servidores web seguros

HTTP (Hypertext Transfer Protocol = Protocolo de Transferencia de Hipertexto). **No es un protocolo seguro**:

- Intercambio de información en texto plano (sniffing).
- No se garantiza que los equipos involucrados en la transferencia sean seguros.
- Robo y falsificación de cookies y/o parámetros (robo de identidad, suplantación de web).
- Vulnerable entre clientes y servidores.
- Vulnerable en las aplicaciones.

Protocolo HTTPS

HTTPS (Hypertext Transfer Protocol Secure= Protocolo de Transferencia de Hipertexto Seguro).

HTTPS es la versión segura de HTTP, tiene el objetivo de que sí se intercepta la información, no pueda verse el contenido.

Criptografía (escritura escondida)

Es la ciencia que estudia la escritura oculta. Se ocupa del cifrado y descifrado de los mensajes.

Cifrar la información consiste en transformar un mensaje en claro en un mensaje que solo pueda ser descifrado por alguien autorizado.

Se basa en algoritmos (públicos) y en claves de cifrado.

Existen dos tipos básicos de algoritmos de cifrado:

- Algoritmos de clave simétrica (secreta).
- Algoritmos de clave asimétrica (pública).

Algoritmos de clave simétrica

- https://www.youtube.com/watch?v=46Pwz2V-t8Q
- criptografía algoritmos de clave simétrica

Algoritmos de clave pública

- https://www.youtube.com/watch?v=On1clzor4x4
- Cifrado de clave pública

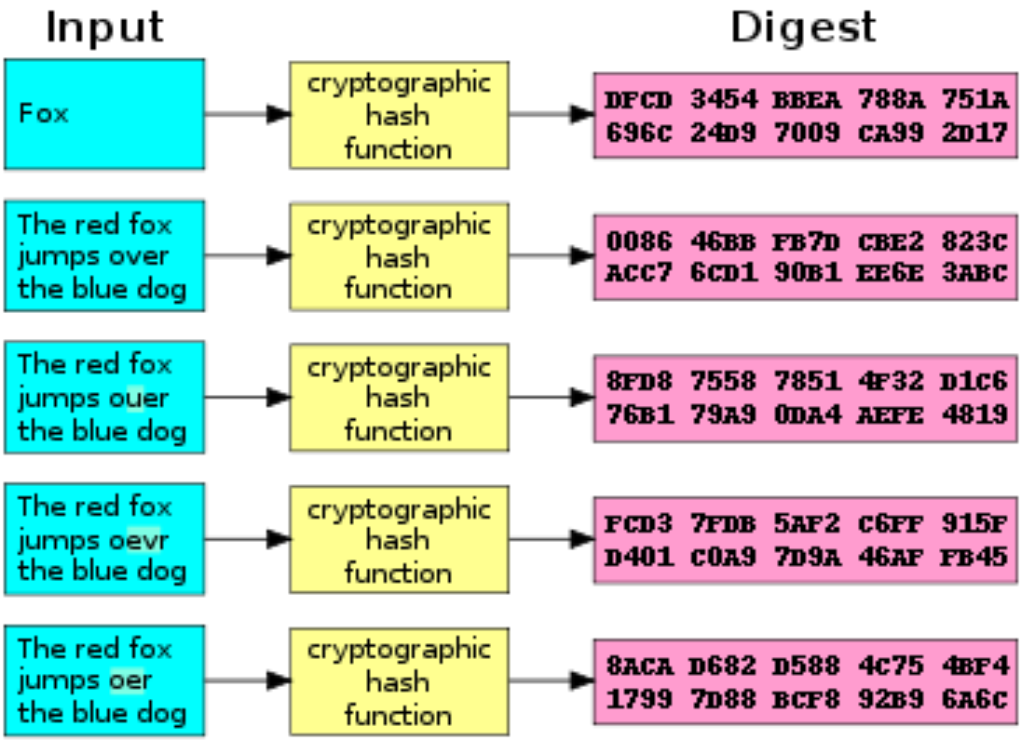
Se basan en el uso de dos claves: una pública y otra privada.

Cada emisor/receptor tiene dos claves:

- La clave privada solo la conoce el dueño de la clave, luego no se publica (no se envía por la red).
- La clave pública es conocida por otros.

Se genera al mismo tiempo, dando lugar a pares biunívocos, de manera que la combinación pública-privada es única.

Algoritmos de autenticación hash. Funciones hash



Función hash = método matemático para generar claves o llaves que representen de forma casi unívoca a un conjunto de datos. La operación ser realiza sobre el conjunto de datos de cualquier longitud y su salida será una huella digital de tamaño fijo e independiente del documento original, de forma que no sea legible.

Prueba la conversión hash on-line: http://sha1-hash-online.waraxe.us/

Requisitos que debe cumplir una función hash:

- Que sea imposible obtener el texto original a partir de la huella digital.
- Que se imposible encontrar un conjunto de datos diferentes que tengan la misma huella (aunque en algunos casos particulares este requisito puede no cumplirse debido al tamaño de 160 bits SHA-1, ocasiona que haya la misma huella para distintos documentos).
- Que pueda transformar un texto de longitud variable en una huella de tamaño fijo (SHA-1 es de 160 bits).
- Que sea fácil de usar e implementar.

A partir de un hash o huella digital no podemos recuperar el conjunto de datos originales. Los más conocidos son el MD5 y el SHA-1.

Cifrar una huella digital se conoce como firma digital.

Ejemplos de funciones hash:

- **MD5**. (Message-Digest Algorithm 5 - Algoritmo de Resumen del Mensaje 5). Es un algoritmo **criptográfico de 128 bits** y es uno de los más usados actualmente.

- **SHA-1** (Secure Hash Algorithm - Algoritmo de Hash Seguro). Es un conjunto de funciones hash y es muy usado en aplicaciones de firma electrónica ya que genera códigos hash entre 160 a 254 bits.
- **SHA-2**. Es uno de los algoritmos más usados en diversos protocolos por su alcance a nivel de seguridad, tales como **TLS, SSL, PGP, SSH**, entre otros. SHA-256 siempre cuenta con 64 caracteres independiente del tamaño de la longitud del archivo donde cada digito es una letra de la A hasta la F y cada uno representa 4 bits de información por lo cual **64 caracteres x 4 bits= 256 bits** lo cual aumenta su nivel de protección.

Software, por ejemplo:

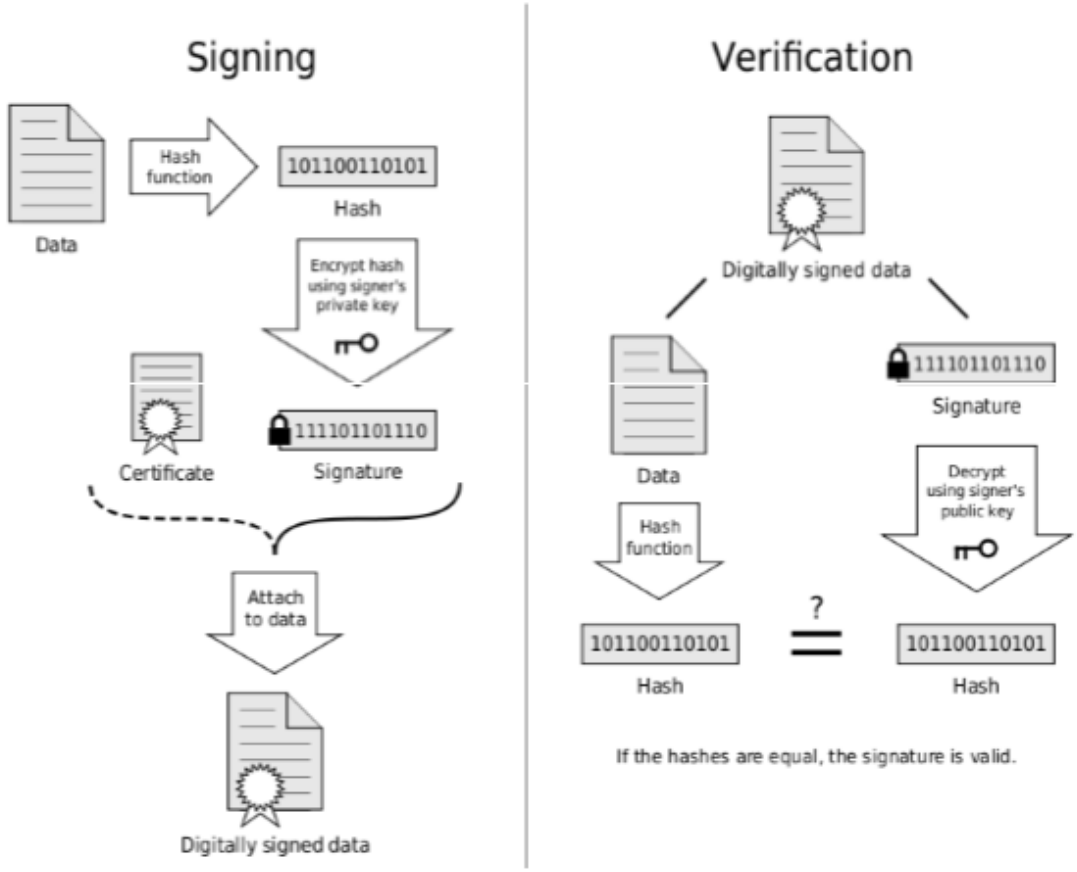
- GNU/Linux: md5sum, sha-1, sha-256 sum (por defecto instalados).
- Windows: Snap MD5 (MD5, sha-1, SHA-256).

Firma digital

Permite firmar un documento digitalmente, de manera que garantizaremos:

- Integridad: Le dota de veracidad, garantizando que el mensaje no ha sido modificado y que se respeta su integridad.
- No repudio: El usuario que lo ha firmado no puede repudiarlo.

La firma digital se basa en algoritmos de clave pública y en funciones resumen hash



Firma:

- Se calcula el resumen hash del documento.
- El resumen se cifra con la clave privada del usuario. Así se asegura que el único que ha firmado el documento es el usuario, porque es el único que conoce la clave privada.
- El resultado se conoce como firma digital del documento.

Verificación:

- La firma se descifra usando la clave pública del usuario (cualquiera la puede tener, por lo tanto, cualquiera puede verificarla).
- Se obtienen el valor resumen del documento firmado.
- Se comparan los dos resúmenes y si coinciden la firma es válida.

Más sobre firma digital : http://www.cert.fnmt.es/curso-de-criptografia/criptografia-de-clave-asimetrica/firma-digital

Certificado digital

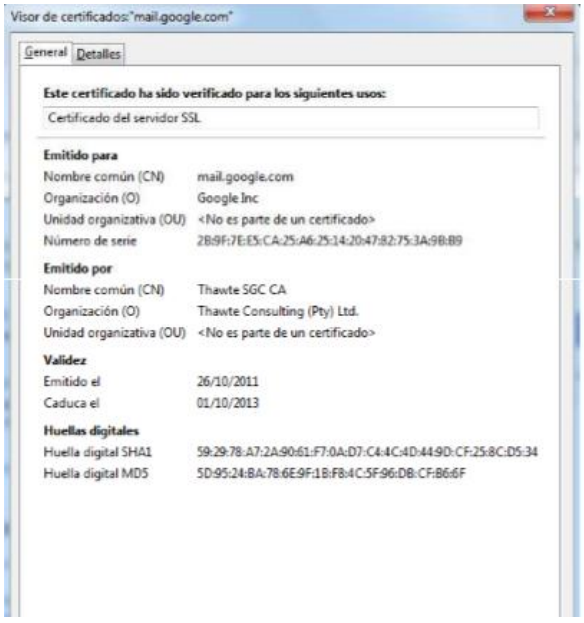
El Certificado Digital es un documento que contiene:

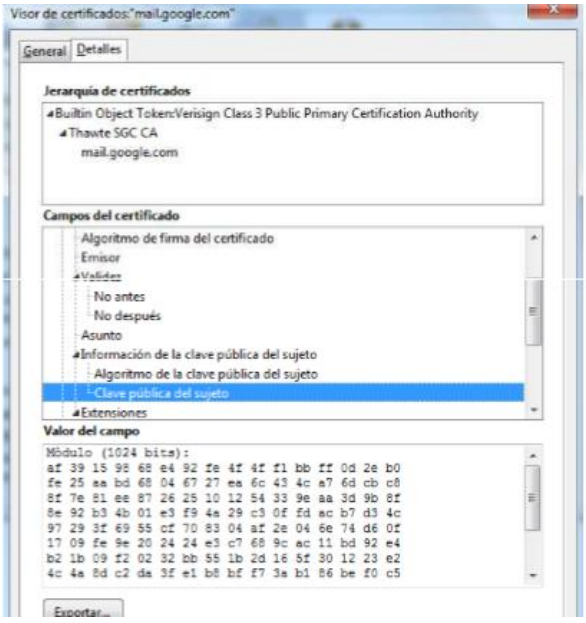
- Información sobre una persona, entidad, empresa, organización, etc.
- La clave pública del propietario (persona, entidad, empresa, etc.)
- La firma digital de un organismo de confianza, una autoridad de certificación (CA, Certificate Authority) que garantiza que la clave pública que contiene el certificado se corresponde con el propietario del mismo.

Más sobre certificados digitales: http://www.cert.fnmt.es/curso-de-criptografia/certificados-digitales

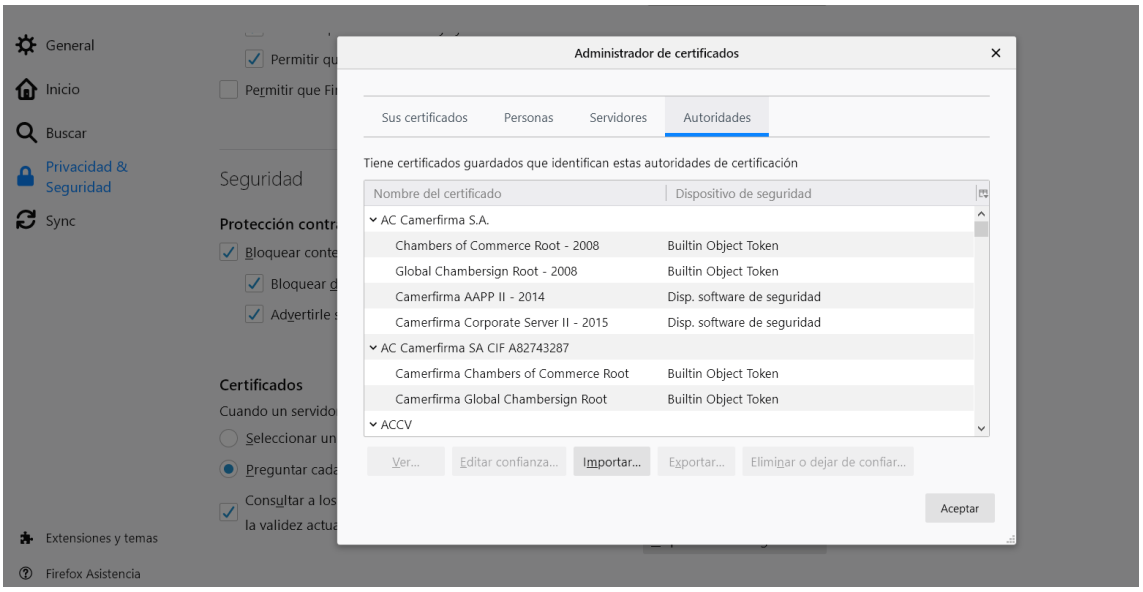
Formatos de los certificados

- **Formato X.509**.





- **Certificados raíz**
  - Emitidos por autoridades de certificación para sí mismas con su clave pública.
  - Son necesarios para verificar la autenticidad de los certificados que emiten.



- **Certificados auto-firmados:**
  - Es el que se realiza sin la intervención de una CA
  - No existe ningún mecanismo automático que garantice su autenticidad.

Los certificados digitales son útiles para:

- Verificar que nos conectamos con el servidor que deseamos y no con un impostor.
- El servidor sepa que cuando nos conectamos con él somos nosotros y no otra identidad.

Por eso, en cualquier entorno de clave pública, es necesaria una **Tercera Parte Confiable (TPC o TTP, Trusted Third Part)**.

La TPC avalará que el certificado es de fiar mediante su firma digital sobre el certificado.

La TPC que se encarga de la firma digital de los certificados de los usuarios en un entorno de clave pública, se conoce como Autoridad de Certificación (AC).

La **PKI (Public Key Infrastructures)** es el modelo de confianza basado en TPC. Se trata de un conjunto de protocolos, servicios y estándares que soportan las aplicaciones basadas en criptografía de clave pública.

Algunos de los servicios ofrecidos por una ICP (PKI) son los siguientes:

- Registro de claves: emisión de un nuevo certificado para una clave pública.
- Revocación de certificados: cancelación de un certificado previamente emitido.
- Selección de claves: publicación de la clave pública de los usuarios.
- Evaluación de la confianza: determinación sobre si un certificado es válido y qué operaciones están permitidas para dicho certificado.

- Recuperación de claves: posibilidad de recuperar las claves de un usuario.

Las ICP (PKI) están compuestas por:

- Autoridad de Certificación (AC): realiza la firma de los certificados con su clave privada y gestiona la lista de certificados revocados.
- Autoridad de Registro (AR): es la interfaz hacia el mundo exterior. Recibe las solicitudes de los certificados y revocaciones, comprueba los datos de los sujetos que hacen las peticiones y traslada los certificados y revocaciones a la AC para que los firme.

4. SSL y Apache

SSL son las siglas en inglés de Secure Socket Layer (en español capa de conexión segura).

Este protocolo ha sido sucedido por TLS, que son las siglas en inglés de Transport Layer Security (en español seguridad de la capa de transporte). Versiones de TLS tienen un equivalente en SSL, por ejemplo, TLS 1.2 corresponde a SSL 3.3; de ahí que aún sea común que se refiera a este protocolo como SSL.

El método de cifrado SSL/TLS utiliza un método de cifrado de clave pública (cifrado asimétrico) para la autenticación del servidor.

Características SSL/TSL:

- Se ejecutan en una capa entre los protocolos de aplicación (http, smtp o ftp) y el protocolo de transporte TCP.
- HTTPS, FTPS, SMTPS, POPS, IMAPS, etc., se basan en SSL/TSL.
- También es posible implementarlo sobre UDP.
- Configuración habitual:
    - El servidor de la comunicación tiene que estar autenticado.
    - El servidor emite el Certificado digital.

Introducción al protocolo SSL

Aplicaciones SSL/TLS:

- Comercio electrónico en Internet -> HTTPS
- Correo electrónico seguro -> SMTPS, IMAPS, POPS.
- Autenticación y cifrado en tráfico de voz (VoIp).

En Apache, el módulo SSL no viene activado, así que si queremos utilizarlo tendremos que activarlo.

5. Acceso a carpetas seguras

En ocasiones entre el servidor y el navegador debe viajar información sensible, que debe ser cifrada. Se trata del **cifrado de clave público o asimétrico**: clave pública (**kpub**) y clave privada (**kpriv**).

- La kpub suele publicarse para que sea conocida por cualquiera.
- La kpriv no interesa que sea conocida, tan solo es para el propietario de la misma.

Tanto kpub como kpriv son necesarias para que la comunicación sea posible. Así:

- La información cifrada mediante kpub solo puede ser descifrada mediante kpriv.
- La información cifrada mediante kpriv solo puede ser descifrada mediante kpub.

Apache puede configurarse como AC (Autoridad de Certificación), pero los navegadores pueden desconfiar de los certificados creados.

HTTPS:

Para asegurar la información, el protocolo HTTPS utiliza certificados, que cuando se validan transfieren la información cifrada.

A la hora de cifrar la información, debemos tener en cuenta que es un proceso que consume tiempo de computación, así que debemos decidir:

- Qué parte de la información transferida entre cliente y servidor tiene que ir cifrada y cual no. Por ejemplo, solamente es necesario que sea cifrada la autenticación a dicha información.

- Si tenemos que configurar el servidor para que la información esté cifrada en todo el dominio o solo en el intento de acceso a la misma.

Apache Web Server puede emitir certificados, pero en algún navegador pueden ser interpretados como amenazas o peligros.

¿Por qué un Certificado puede interpretarse como una amenaza?

¿Confiarías en un DNI que no estuviese certificado por una entidad de confianza como el Ministerio de Interior?

Lo mismo sucede con los navegadores, solo confían en los certificados emitidos por su lista de **Entidades Certificadoras**, donde verifican, autentican y dan validez a los certificados.

Por eso, aunque podamos crear nuestros certificados en un servidor web, si salimos a Internet pueden ser interpretados como peligrosos por el navegador, que alertará con un aviso de problema de seguridad.

¿Cómo funciona https?

- HTTPS utiliza el puerto 443, en vez del puerto 80 (http).

- HTTPS cifra la información mediante SSL (Secure Socket Layers) y así crear un canal de transferencia de cifrado.

SSL es un protocolo que proporciona privacidad e integridad entre dos aplicaciones utilizando http.

¿Cómo se transfiere información?

1. Con http → Escribo la URL con http://

- Se traduce el dominio DNS a IP.
- Se busca en la IP obtenida la página solicitada por el puerto 80 (puerto TCP asignado por defecto al protocolo HTTP).
- Si el servidor tiene la página, la ofrece al navegador.

2. Con https → Escribo la URL https://

- Se traduce el dominio DNS a IP.
- Se busca en la IP obtenida la página solicitada por el puerto 443 (puerto TCP asignado por defecto al protocolo HTTPS).
- Se inicia negociación SSL, entre otras cosas, el servidor envía su certificado.
- Si el certificado es firmado por una Entidad Certificadora de confianza, acepta el certificado y cifra la comunicación.
- Se transfiere la página web de forma cifrada.

Protocolos y puertos:

Normalmente el servidor web espera el protocolo HTTP a través del puerto 80 y HTTPS a través del puerto 443.

Los puertos se pueden cambiar y configurar en el servidor web, pero cualquiera que quiera acceder a esas páginas debe saber el puerto TCP de la solicitud. Por ejemplo:

- http://www.miweb.local:8080 → El servidor web espera a miweb.local en el puerto 8080.
- https://www.miweb.local:4333 → El servidor web espera a miweb.local en el puerto 4333.