



6

RESUMEN UNIDAD DIDÁCTICA: SERVICIO WEB (HTTP)

Contenido

1. LA WORLD WIDE WEB	3
1.1. ORIGEN	3
1.2. CONCEPTOS	4
2. EL PROTOCOLO HTTP.....	6
2.1. INTRODUCCIÓN	6
2.2. FUNCIONAMIENTO DE LA WEB	6
2.3. LAS COOKIES	8
2.4. CÓDIGOS DE ERROR.....	8
2.5. TIPOS MIME	10
2.6. FORMATO DE LOS MENSAJES	11
3. LOS NAVEGADORES.....	14
3.1. INTRODUCCIÓN	14
3.2. OPCIONES DE CONFIGURACIÓN	15
4. SERVIDOR WEB EN GNU/LINUX	16
4.1. INSTALACIÓN DE APACHE HTTP SERVER	16
4.2. CONFIGURACIÓN BÁSICA.....	16
4.3. OTROS PARÁMETROS	20
5. PREPARAR UN SERVIDOR WEB SEGURO	21
5.1. PROTOCOLO HTTPS.....	21
5.2. CERTIFICADO DIGITAL	23
5.3. SERVIDOR WEB APACHE CON HTTPS	24

SERVICIO WEB (HTTP)

Ciclo Formativo de Grado Medio **Sistemas Microinformáticos y Redes**
Módulo: **Servicios en Red**

José Manuel Martínez del Hoyo Cañizares

I.E.S. Maestre de Calatrava. Ciudad Real.
<http://iesmaestredecalatrava.es>

1. La World Wide Web

1.1. Origen

La **World Wide Web** (*la “telaraña mundial”*) o simplemente la **Web**, surge en 1990 como resultado de un proyecto realizado en el instituto **CERN** de Ginebra (Suiza).

El proyecto WWW, dirigido por **Tim Berners Lee**, tuvo como objetivo crear un sistema capaz de permitir a científicos, no informáticos, **compartir información de manera sencilla en Internet**. Después de tres años, el uso del sistema se había extendido a muchas instituciones y en 1993 se había impuesto como herramienta para obtener y compartir información por medio de Internet.

La Web es el servicio de Internet más reciente y espectacular, y desde su origen hasta hoy día, no ha cesado de desarrollarse y evolucionar con nuevas capacidades, hacia nuevos servicios y utilidades.



Cómo se originó la Web

http://www.hipertexto.info/documentos/h_www.htm



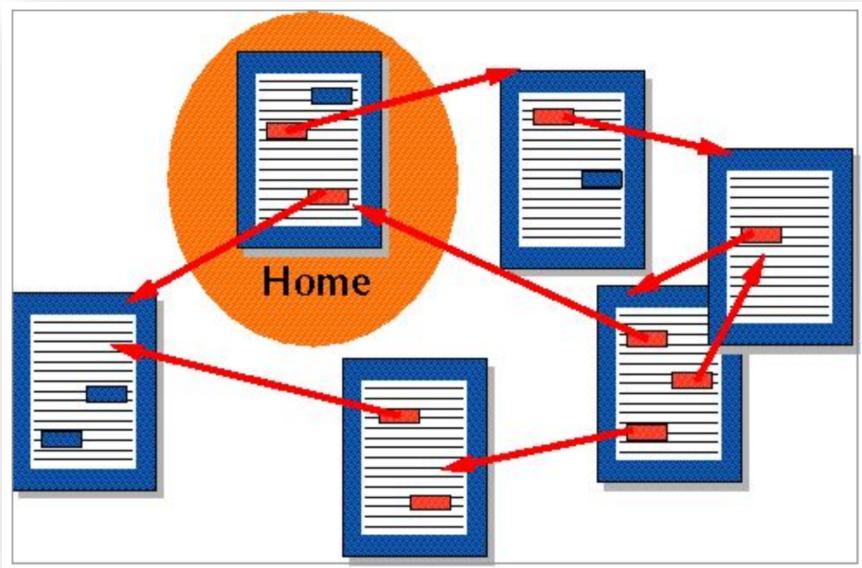
La primera página Web de la historia

<http://www.w3.org/History/19921103-hypertext/hypertext/WWW/TheProject.html>

1.2. Conceptos

El servicio Web está basado en una **RED GLOBAL de DOCUMENTOS** con las siguientes características:

- ✓ **Documentos Multimedia**: integran imágenes, vídeo, texto, gráficos, etc. y son accesibles de mediante una **Interfaz Gráfica**.
- ✓ Son accesibles en **Internet**.
- ✓ Tiene estructura de **Hipertexto**: los documentos están enlazados entre sí, localizados en diferentes ubicaciones de la red.



Hipertexto

- El concepto de hipertexto no es algo nuevo que surgiera con la Web
- Se denomina hipertexto al sistema que permite **saltar de un punto a otro mediante palabras de enlace**.
- En la web, este sistema funciona de manera que en un documento de texto escrito en HTML, una palabra, llamada **enlace hipertexto (hyperlink o link)** nos lleva a otra página HTML que puede residir en el mismo servidor web o en cualquier otro servidor de Internet.

- Por defecto, los navegadores resaltan estos *hiper-enlaces* con otro color y los subrayan (generalmente los veremos en azul): [enlace](#); después de ser visitados podrán cambiar de color, por ejemplo a morado: [enlace](#).
- El enlace puede llevarnos a **otro tipo de recurso** que no sea otra página HTML, como por ejemplo una imagen JPEG, un documento PDF, una canción MP3, un video o un sitio FTP.

Multimedia

- Ya desde los comienzos de la web, los **navegadores**, aparte de interpretar el código HTML de las páginas y realizar su presentación en pantalla, eran capaces de visualizar por ellos mismos **imágenes JPEG y GIF**.
- Hoy día, son capaces de representar una gran cantidad de formatos de imagen, video o sonido, por ellos mismos o por plugins instalados.
- Aunque han existido navegadores en modo texto como **Lynx** (1992), el primer navegador creado y utilizado en el proyecto WWW, funcionaba en **Modo Gráfico** en estaciones **NeXT**.
- Poco después, el navegador **Mosaic** funcionó en sistemas UNIX sobre el sistema gráfico X-Window.

La Web e Internet

- Un aspecto importante es que la **WWW no es igual a Internet**. Aunque una de las formas de intercambiar información más utilizada en Internet es la Web (HTTP), la web es únicamente un servicio más de Internet.
- Recuerda que el correo electrónico es otro servicio diferente y casi tan usado como la propia Web. Pero al ser posible su acceso desde un navegador (WebMail), las personas no suelen distinguir que es un servicio diferente.

2. El protocolo HTTP

2.1. Introducción

HTTP es el **Protocolo de Transferencia de Hipertexto** (*Hypertext Transfer Protocol*), protocolo utilizado en cada transacción entre un **servidor HTTP** (servidor web), y un **cliente HTTP** (navegador).

Este protocolo se describió en el RFC 1945 (HTTP/1.0, año 1996), que ha sido ampliado y modificado hasta llegar al RFC 7540 (2015) que define la versión **HTTP 1.2**.

El puerto estándar para este protocolo es el **80**.

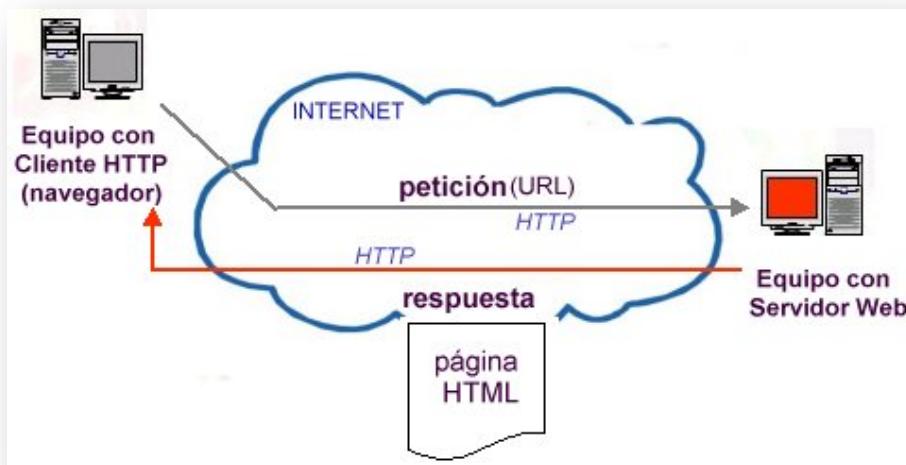
2.2. Funcionamiento de la web

La Web, al igual que el resto de servicios de Internet, funciona siguiendo el **modelo cliente-servidor** en una red TCP/IP, ya sea local o interconectada con las demás redes a través de Internet.

Un equipo **cliente**, mediante un programa denominado **navegador** (*web browser*), realiza la petición mediante HTTP a un **servidor Web** solicitándole una página HTML. Una vez recibida, el navegador la interpreta y la visualiza.

- Las páginas que recibe el navegador son documentos **HTML**, que el navegador interpretará junto con los ficheros **CSS**, para poder mostrarlo al usuario en el formato adecuado.
- Si lo que recibe no es un documento de texto, sino un **objeto multimedia** (video, sonido, etc.) no reconocido por el navegador, éste deberá activar una aplicación externa capaz de gestionarlo, o preguntar al usuario qué desea hacer con él.
- Hay algunos formatos (aparte del texto plano y HTML) reconocibles por el propio navegador sin necesidad de aplicación externa, por ejemplo los formatos de imágenes JPEG, GIF y PNG.

Diálogo entre navegador y servidor web



Desde el punto de vista del **Servidor**, su función es atender peticiones de los clientes HTTP, que solicitan páginas y otros documentos.

Si el servidor encuentra la página HTML o el documento solicitado, lo envía. En caso de no encontrarlo, se envía un código de error.

Es importante comprender que el protocolo HTTP **es un protocolo sin estado**, es decir, no recuerda ningún suceso de conexiones anteriores a la actual.

Esto significa que en cada petición de un cliente,

- se establece la conexión,
- se devuelve el documento (o el error) y
- se cierra la conexión, por tanto no recordará nada acerca de ese cliente si poco después realiza una nueva petición.

2.3. Las cookies

Para resolver estas situaciones de “falta de memoria” del protocolo HTTP, y poder recordar la información de la sesión actual, se utilizan las denominadas **“cookies”**.

Las cookies son simples **ficheros de texto** que se intercambian entre servidor y cliente.

Realmente es **el servidor el que pide a nuestro navegador que escriba ese fichero en el disco**, con información sobre la visita a su sitio web, de modo que la próxima vez que entremos en ese sitio, el servidor sabrá alguna información de sesiones anteriores gracias a estas cookies.

Un ejemplo podría ser el idioma seleccionado para ver un determinado sitio web. Una vez seleccionado la primera vez, cuando nos conectemos desde el mismo equipo, no habrá que seleccionarlo otra vez.

En resumen, las cookies constituyen una potente herramienta empleada por los servidores Web para almacenar y recuperar **información acerca de sus visitantes**.

2.4. Códigos de error

Los códigos de error o de estado, son números de tres cifras que indican la respuesta del servidor web a una determinada petición.

El **primer dígito** del código de respuesta especifica una de las cinco clases de respuesta:

- 1xx: Respuestas informativas
- 2xx: Peticiones correctas
- 3xx: Redirecciones
- 4xx: Errores del cliente
- 5xx: Errores del servidor

Los principales códigos son:

- **100: Continua** (continuar con la petición).
- **200: OK** (petición correcta).
- **206: Contenido parcial.** La petición servirá parcialmente el contenido solicitado. Esta característica es utilizada por herramientas de descarga como wget para continuar la transferencia de descargas anteriormente interrumpidas, o para dividir una descarga y procesar las partes simultáneamente.
- **301: Movido permanentemente.**
- **307: Redirección temporal.**
- **401: No autorizado.** La autentificación ha fallado.
- **403: Prohibido (Forbidden).** La solicitud fue legal, pero el servidor rehúsa a responderla.
- **404: No encontrado (Not Found).** Se utiliza cuando el servidor web no encuentra la página o recurso solicitado.



- **408: Tiempo de espera agotado.**
- **500: Error interno (Internal Server Error).**
- **503: Servicio no disponible (Service Unavailable).**

2.5. Tipos MIME

Los tipos **MIME** (*Multipart Internet Mail Extension*) fueron en un primer momento utilizados para extender las características del correo electrónico, pero hoy su uso se ha extendido, siendo también denominados IMT (*Internet Media Types*).

El registro de los tipos MIME lo controla la **IANA** (*Internet Assigned Numbers Authority*), y en su sitio web podemos obtener la lista completa y actualizada de los tipos registrados.

Los tipos MIME se componen de un **tipo** y un **subtipo**.

Ejemplos

- Una página web que es un fichero de texto escrito en HTML, tendrá como tipo MIME **text/html**.
- Una foto en formato JPEG tiene el tipo MIME **image/jpeg**.

El protocolo **HTTP usa tipos MIME** en sus **encabezados** para realizar las siguientes funciones:

- a) Permitir la **negociación del contenido**.

El cliente, en su petición, incluye los tipos MIME que acepta.

Ejemplo

Si un navegador puede soportar documentos comprimidos de tipo **“application/zip”**, lo indica con el encabezado http mediante la expresión **Allow: “application/zip”**.

- b) Informar al cliente (navegador) del tipo de datos que está recibiendo del servidor.

Esto se hace con el encabezado **Content-Type**.

Un navegador típico puede manejar los datos de tres maneras distintas según el tipo MIME indicado en el Content-Type:

- Si el tipo MIME es “**text/html**”, visualiza el documento.
- Si el tipo MIME es “**application/....**”, debe llamar a una aplicación externa. Esta aplicación dependerá del subtipo.

Ejemplo

Content-Type: “application/pdf”.

El navegador deberá llamar a la aplicación que maneja ficheros PDF que es Adobe Reader. Si no dispone de la aplicación adecuada al subtipo, preguntará al usuario si quiere guardar el fichero.

- Si el tipo MIME no es entendible por el navegador preguntará qué hacer (por ejemplo si desea guardar el fichero).

2.6. Formato de los mensajes

En las transacciones HTTP, el cliente envía un **mensaje al servidor** (petición). Este mensaje contiene una cabecera y, opcionalmente, algún dato.

La cabecera contiene la **línea de petición** o *request*, y los **parámetros** de petición.

La **línea de petición** está compuesta por:

- método que será **GET** en la petición
- URL relativa, por ejemplo **index.html**
- la versión del protocolo, por ejemplo **HTTP/1.1**.

Cada **parámetro** de petición va seguido de dos puntos (:) y el valor.

Algunos de estos parámetros son:

- **Date:** Tue, 18 Jan 2018 08:12:31 GMT+1 (fecha y hora en inglés)
- **Host:** www.example.com (nombre del servidor web)
- **User-Agent:** IE 11 (navegador utilizado)

Ejemplo diálogo HTTP

Obtener el recurso con URL <http://www.example.com/index.html>

- **Petición:** el navegador abre una conexión con el host www.example.com, puerto 80 que es el puerto por defecto para HTTP.
- Envía un mensaje HTTP, tipo GET:

```
GET /index.html HTTP/1.1
Host: www.example.com
User-Agent: IE 11
[Línea en blanco]
```

- El **servidor acepta la petición** y busca el recurso solicitado en su disco, en este caso el fichero **index.html**.
- **Respuesta:** la respuesta del servidor está formada por encabezados seguidos del recurso solicitado, en el caso de una página web. La respuesta tiene un código 200.

HTTP/1.1 200 OK

Date: Fri, 31 Dec 2018 23:59:59 GMT
Server: Apache/2.1.1.2 (Unix)
Last-Modified: Wed, 08 Jan 2018 23:11:55 GMT
Content-Type: text/html
Content-Length: 1221

Encabezado

```
<html>
<head><title>Ejemplo</title></head>
<body>
    <h1>Principal</h1>
    <hr>
    <h3>www.example.com</h3>
</body>
</html>
```

Recurso:
página HTML

En caso de NO haber encontrado el recurso solicitado, la respuesta tendría un código 404 (Not Found).

- **Presentación:** el navegador recibe el encabezado y el recurso (index.html). Puesto que el código de error es OK, procede a examinar los recursos recibidos.
- Se compone de un único fichero de tipo HTML (tipo MIME text/html) por lo que procede a interpretarlo, realizando la presentación del contenido en su ventana.



3. Los navegadores

3.1. Introducción

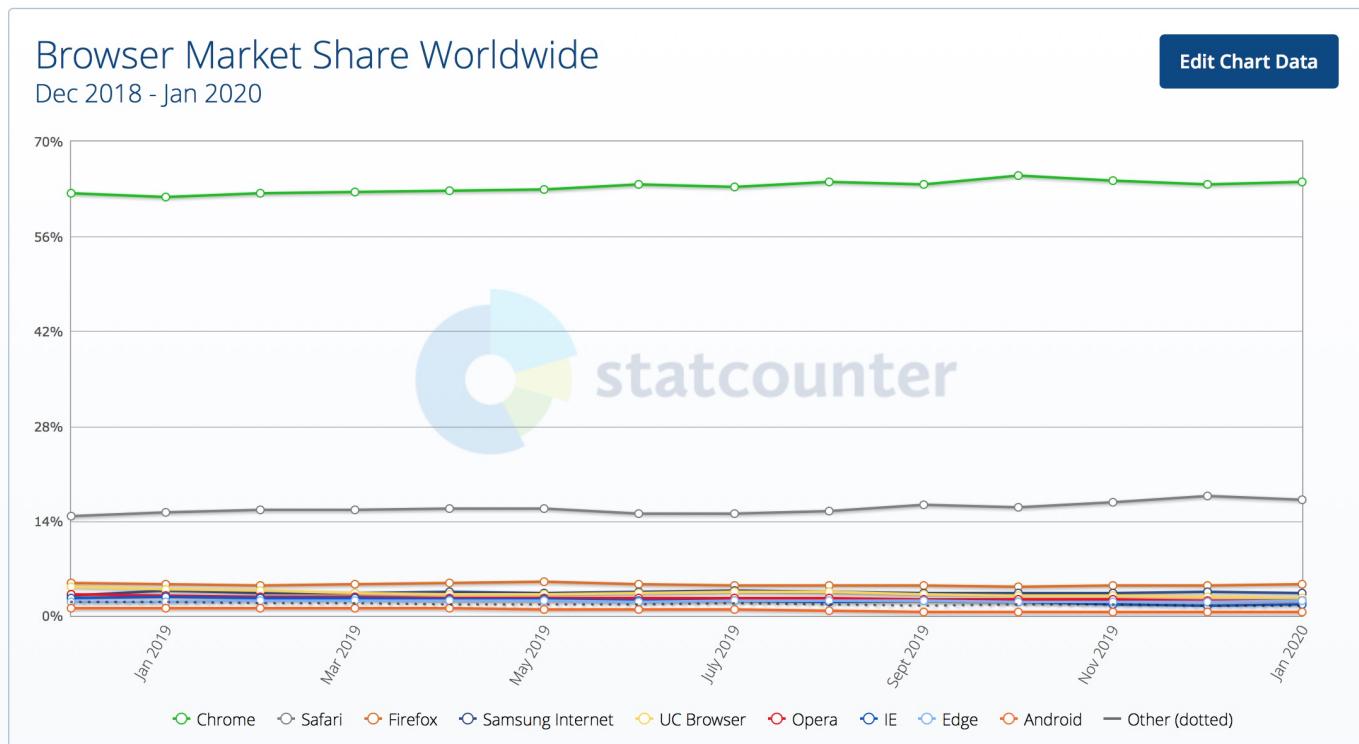
El navegador o **Web browser**, es el programa cliente del servicio web. Hasta hace unos años era el único modo de acceder a la web.

Con la proliferación de las **Apps** en dispositivos móviles, el acceso a la web se ha hecho transparente al usuario, puesto que estas aplicaciones acceden a la Web pero sin que el usuario tenga conciencia exacta de dónde accede como sucede cuando usamos un navegador.

Los navegadores más utilizados son:

- **Chrome** (Google)
- **Safari** (Apple)
- **Firefox**
- **Samsung Internet** (Samsung)
- **UC Browser** (UC Mobile)
- **Opera**
- **Internet Explorer** (Microsoft)

Estadísticas de uso (2019). Fuente: <http://gs.statcounter.com>



3.2. Opciones de configuración

Todos los navegadores poseen unas opciones de configuración que suelen prácticamente las mismas en todos ellos:

Seguridad

Aceptación o no de **cookies**, ejecución de **Java**, **JavaScript**, etc.

Navegación

Contraseñas guardadas en caché, **autorelleno**, etc.

Historial

Sitios visitados

HTTPS

Administración de **certificados digitales**

Conexión red

Uso de servidores **proxy**.

Apariencia:

Font por defecto, tamaño, etc.

4. Servidor web en GNU/Linux

4.1. Instalación de Apache HTTP Server

Para **obtener Apache2 e instalarlo** en el sistema, puedes usar el

- Gestor de Paquetes **Synaptic** localizado en:
Menú > Administración > Gestor de Paquetes Synaptic
- o, mejor, el sistema **APT** de Debian.
Ejecuta el siguiente comando (*apt-get*) como superusuario:

```
$ sudo apt-get install apache2
```

Si no encuentra el paquete a instalar, no olvides teclear el siguiente comando:

```
$ sudo apt-get update
```



Sitio oficial del proyecto Apache HTTP Server

https://projects.apache.org/project.html?httpd-http_server

4.2. Configuración básica

Primero, debes asegurarte que tu máquina tiene una dirección IP válida y que es estática.

Los **ficheros de configuración** del servicio están en el directorio /etc/apache2:



/etc/apache2/apache2.conf

/etc/apache2/ports.conf

...

- **apache2.conf**: Configuración Global del servicio
- **ports.conf**: Configuración del puerto
- etc.

Además, CADA SITIO WEB, tendrá su fichero de configuración en:



```
/etc/apache2/sites-available/  
    000-default.conf  
    aulasmr.es.conf  
    ...
```

**Configuración del sitio web al que se accederá con el nombre
www.aulasmr.es**

Pasos:

- 1) Crea un directorio para el o los sitios web que alojará el servidor web. Por ejemplo **websites**, dentro del directorio de usuario. (también puedes usar el directorio por defecto /var/www).

Una buena práctica es nombrar al directorio con el nombre de la zona del sitio web. Por ejemplo **aulasmr.es**

```
$ cd /home/alumno  
$ mkdir websites  
$ cd websites  
$ mkdir aulasmr.es
```

- 2) Crea un fichero HTML de prueba como página principal. Llámalo **index.html**

```
$ cd aulasmr.es  
$ nano index.html
```

El fichero podría tener simplemente la siguiente linea:

```
<h1>Funciona</h1>
```

- 3) Crea una **copia** del fichero **000-default.conf** y llámalo igual que el nombre de zona del sitio web, seguido de **.conf**.

Importante: estos ficheros DEBEN acabar en .conf

```
$ cd /etc/apache2/sites-available  
$ sudo cp 000-default.conf aulasmr.es.conf
```

- 4) Configura la localización del contenido para el nuevo sitio web. Para ello debes editar el fichero **aulasmr.es.conf** y modificar el parámetro **DocumentRoot**.

```
# DocumentRoot /var/www  
DocumentRoot /home/alumno/websites/aulasmr.es
```

- 5) El acceso por un navegador al sitio web, se hará por nombre. Además, será obligatorio **utilizar nombres si el servidor mantiene varios sitios web**. Será la forma que tiene el servidor web de identificar qué sitio está pidiendo el navegador.

Para configurar el nombre, debes editar el fichero **aulasmr.es.conf** y modificar el parámetro **ServerName**.

```
ServerName www.aulasmr.es
```

- 6) Añade cláusula **Directory** para el directorio donde está alojado el contenido. Para ellos debes editar el fichero **apache2.conf**.

```
<Directory /home/alumno/websites/aulasmr.es>  
    Options Indexes FollowSymLinks  
    AllowOverride None  
    Require all granted  
</Directory>
```

- 7) Una vez realizada la configuración básica del sitio web, debes **habilitarlo** con el comando **a2ensite**.

```
$ cd /etc/apache2/sites-available  
$ sudo a2ensite aulasmr.es.conf
```

También podrías **deshabilitar el sitio por defecto** con el comando **a2dissite**.

```
$ cd /etc/apache2/sites-available  
$ sudo a2dissite 000-default.conf
```

- 8) Podrás ver los sitios habilitados en el directorio **sites-enabled**.

```
$ cd /etc/apache2/sites-enabled  
$ ls -l  
→ aulasmr.es.conf
```

- 9) Comprueba que no has cometido ningún error al escribir la configuración. Para ello utiliza el siguiente comando:

```
$ sudo apachectl configtest  
→ Syntax OK
```

- 10) Finalmente, reinicia el servicio o recarga la nueva configuración con el comando **service**.

```
$ sudo service apache2 restart
```

o simplemente **reload** (para recargar la configuración)

```
$ sudo service apache2 reload
```

No olvides verificar que el servicio está en ejecución con el comando **ps**.

```
$ sudo ps -ef | grep apache
```

Prueba desde un cliente

- 11) Verifica que el cliente está usando un servidor DNS donde esté la zona registrada

Por ejemplo: www.aulasmr.es

- Zona aulasmr.es >
registro A para **www** con la dirección IP del servidor

- 12) Teclea el siguiente URL: <http://www.aulasmr.es>

4.3. Otros parámetros

ServerAdmin

Indica la dirección de correo electrónico del administrador del servidor web.

ErrorDocument

Indica la página HTML que enviará Apache cuando se produzca un error determinado.

Ejemplo:

- Cuando un URL indica un fichero que el servidor web no encuentra, se produce una respuesta con **código 404 File Not Found** y nos llega la página por defecto que tiene Apache para ese error.
- Con la siguiente linea en el **fichero de configuración del sitio web**, cuando se produzca un 404, enviará la página con nombre **e404.html** localizada en el directorio raíz del sitio web:

```
ErrorDocument 404 /e404.html
```

DirectoryIndex

Indica cuáles son los **ficheros por defecto** y en qué orden deben buscarse, cuando en el URL se **omite el nombre del fichero**.

Ejemplo:

- Si la página inicial del sitio web es inicio.html y tiene otra llamada default.htm, al teclear en el navegador **http://www.aulasmr.es/**, mostrará la página **inicio.html**:

```
DirectoryIndex index.html inicio.html index.php  
default.htm
```

5. Preparar un servidor web seguro

5.1. Protocolo HTTPS

La comunicación entre un navegador y un servidor **Web** se realiza mediante el protocolo **HTTP**.

Cuando el tráfico en la Web contiene **información sensible y confidencial** (usuarios, claves, datos de pago, etc.), este tipo de comunicación requiere de **cifrado** para impedir que un posible atacante que obtenga esta información.

La versión segura de este protocolo se llama **HTTPS**:

HTTPS = HTTP + SSL/TLS.

- El número de puerto estándar de HTTPS es el **443**.
- El protocolo **SSL/TLS** está basado en la aplicación conjunta de Criptografía **Simétrica** y **Asimétrica**, para conseguir un **canal seguro** de comunicación a través de Internet entre el cliente (navegador) y el servidor web de una organización.

De este modo, el posible atacante solo podrá obtener un flujo de datos cifrados.

- Los sistemas criptográficos **simétricos** son el motor principal del **cifrado de datos transferidos** en la comunicación, por la rapidez de operación.
- Los sistemas **asimétricos** se usan para el **intercambio seguro de las claves simétricas**, consiguiendo con ello resolver el problema de la Confidencialidad en la transmisión de datos.
- Además, la **identidad** del servidor web se puede verificar gracias a que su clave pública se presenta dentro de un **Certificado Digital** que debe ser válido gracias a que va **firmado por una Autoridad de Certificación**.

La secuencia es la siguiente:

- El navegador solicita que el servidor web se identifique.
- El servidor web le envía su Certificado Digital SSL/TLS.
- El navegador comprueba la validez del Certificado antes de iniciar el intercambio de datos sensibles, **verificando que la Firma Digital** es de una **Autoridad de Certificación confiable**.
- Si la verificación tiene éxito, es que el servidor web es de la organización de quien dice ser y se puede utilizar su clave pública para cifrar los datos que se envían hacia él.

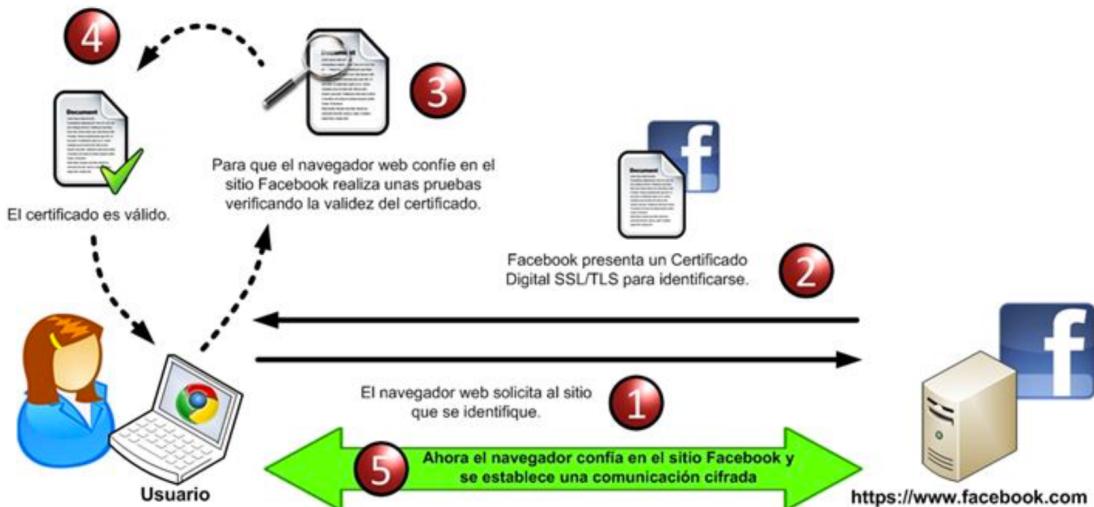
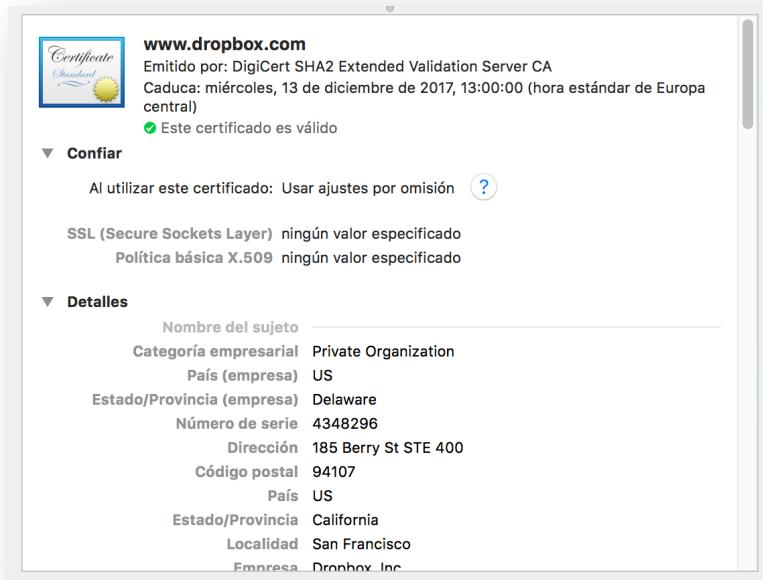


Diagrama 1. Funcionamiento general de SSL/TLS

Todos los navegadores utilizados actualmente, soportan HTTPS. Suelen mostrar un candado en la parte izquierda de la barra de direcciones, para indicar la existencia de un protocolo de comunicaciones seguro, e incluso cambian el color del fondo de la barra de direcciones por azul (Firefox), o verde (Safari, Chrome, Internet Explorer), para identificar un sitio web seguro.





5.2. Certificado digital

Para tener un sitio web seguro, es necesario preparar el servidor web para que acepte conexiones HTTPS.

Para ello, el administrador debe crear un **Certificado digital** para el servidor web.

Este certificado debe estar firmado por una **Autoridad de Certificación (AC)** para que los navegadores al verificar el certificado, lo acepten como válido. La autoridad certifica que el titular del certificado es quien dice ser.

Existe una lista de Autoridades Certificadoras consideradas de confianza e incluidas en la mayoría de los navegadores u otro software que trabaje con certificados. Las Autoridades consideradas de “confianza” son las **AC raíz**, como VeriSign, Thawte y GlobalSign, y también **AC intermedias**, que son firmadas por alguna CA raíz y por tanto también son confiables.

5.3. Servidor web apache con HTTPS

La situación de partida es tener Apache instalado y corriendo en tu distribución GNU/Linux:

- Escucha por el puerto 80 conexiones HTTP.
- Aloja dos sitios web (servidores virtuales).
- Uno de ellos será el sitio de una tienda virtual, que queremos convertir en un **sitio seguro**.

En su fichero de configuración está el parámetro DocumentRoot por ejemplo:

DocumentRoot /home/josema/websites/tiendamaestre.com

- el URL que tecleamos en el navegador para acceder al sitio podría ser por ejemplo **http://www.tiendamaestre.com**
(el DNS utilizado debe tener creada la zona y el registro A correspondiente al servidor web)

PRIMERA PARTE:

Crear certificado y firmarlo por nuestra propia AC

El primer paso es crear el certificado digital y firmarlo por una Autoridad de Certificación. Para realizar la prueba vamos a crear nuestra propia AC.

Esto es posible con el paquete **openssl**.

Tareas

- 1) Realiza todos los pasos como superusuario.

```
$ sudo su  
Password:  
#
```

- 2) El primer requisito es tener instalado el paquete **openssl**.
Se podrá instalar mediante:

```
# apt-get install openssl
```

- 3) Crea Autoridad de Certificación propia:

Preguntará el nombre de la AC (para crear nueva, pulsa Enter).
Teclea también la *Pass Phrase*.
Responde al resto de cuestiones de forma inteligente.

```
# cd /usr/lib/ssl/misc  
# ./CA.sh -newca
```

- 4) Genera la **clave privada** RSA 1024-bit, con algoritmo DES3.
Creará el fichero server.key.

```
# openssl genrsa -des3 -out server.key 1024  
# chmod 600 server.key
```

- 5) Para crear un certificado digital válido, debes generar una **clave pública** y una petición de forma de certificado (*Certificate Signing Request*) para nuestro servidor. Creará el fichero **server.csr**.

```
# openssl req -new -key server.key -out server.csr
```

Pregunta por la *Pass Phrase* de la AC.

Luego pregunta por el *Common Name*: aquí debes teclear el nombre del servidor web: www.tiendamaestre.com

- 6) Ahora crea el certificado digital para nuestro servidor, firmado por nuestra AC. Creará el fichero **server.crt**.

```
# ln -s server.csr newreq.pem  
# ./CA.sh -signreq  
# mv newcert.pem server.crt
```

Pregunta si queremos **firmarla** y hacer **commit**.

Contestar Sí en ambos casos: [y]es.

SEGUNDA PARTE:

Configurar Apache para que use SSL

Tareas

- 1) Crea los subdirectorios **ssl.key** y **ssl.crt** dentro de /etc/apache2.

```
# cd /etc/apache2  
# mkdir ssl.key ssl.crt
```

- 2) Copia el fichero de clave y el certificado en los subdirectorios creados:

```
# cp /usr/lib/ssl/misc/server.key ssl.key  
# cp /usr/lib/ssl/misc/server.crt ssl.crt
```

- 3) Habilita el módulo SSL en Apache:

```
# cd mods-enabled  
# ln -sf ../mods-available/ssl.load ssl.load  
# ln -sf ../mods-available/ssl.conf ssl.conf
```

- 4) Inhabilita el puerto 80 usado por HTTP.
(Otra opción es dejarlo habilitado pero redireccionar las peticiones al 443 de HTTPS).

Fichero **/etc/apache2/ports.conf**

```
#Listen 80
```

- 5) Modifica el fichero de configuración del sitio web:
Fichero **/etc/apache2/sites-available/tiendamaestre.com**

Debe quedar como sigue:

```
<VirtualHost *:443>
    ServerAdmin admin@tiendamaestre.com
    DocumentRoot /home/josema/websites/tiendamaestre.com

    ServerName www.tiendamaestre.com

    SSLEngine On
    SSLCertificateKeyFile      /etc/apache2/ssl.key/server.key
    SSLCertificateFile        /etc/apache2/ssl.crt/server.crt
    ...
    
```

- 6) Chequea el fichero de configuración antes de iniciar Apache:

```
# apache2ctl configtest
Syntax OK
```

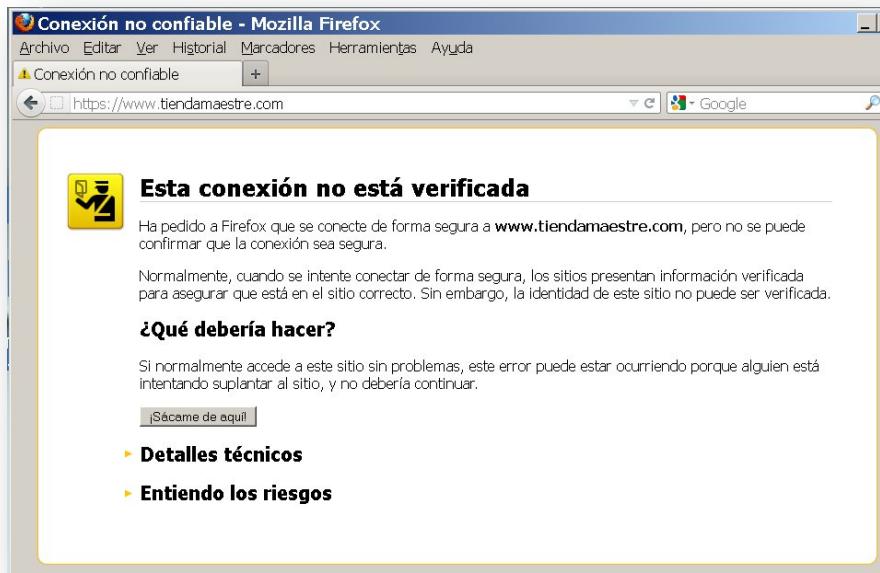
- 7) Inicia el servidor Apache:

```
# service apache2 start

* Starting web server apache2
Apache needs to decrypt your SSL Keys for
www.tiendamaestre.com:443 (RSA)
Please enter passphrase: (teclea la pass phrase)
[OK]
```

- 8) Abre el navegador y accede al sitio web:
<https://www.tiendamaestre.com>

El resultado es frustrante, parece que la prueba ha fallado, pero realmente no es así.



El servidor web acepta la conexión HTTPS, **el canal seguro se ha establecido**, lo único que el navegador nos dice es que el Certificado digital NO es válido.

La razón es simple: el **certificado ha sido firmado por una Autoridad de Certificación desconocida** para el navegador, puesto que la creamos nosotros mismos.

RECUERDA: los navegadores y otro tipo de software que maneje certificados digitales, solo confía en ciertas Autoridades de Certificación.

- 9) Para poder acceder, es posible añadir una Excepción para el sitio desde el navegador: Clic en Entiendo los Riesgos > Añadir Excepción

De este modo somos conscientes de que el sitio tiene un **certificado no verificado** y podría haber suplantación, pero el tráfico entre navegador y el servidor web estará cifrado. Podrá ser una excepción permanente o temporal.



Para ver el certificado del sitio web, solo hay que hacer clic en el botón **Obtener certificado** y luego en botón **Ver**.

Se obtiene la información del Certificado digital del servidor www.tiendamaestre.com (Maestre Ltd), firmado por nuestra AC (IES Maestre de Calatrava Internet AC).



