

Despliegue de Aplicaciones Web

T.2 Los servidores web

Índice

1. La URL
2. Configuración de Apache
3. Módulos: instalación, configuración y uso
4. Servidores virtuales
5. Autenticación y control de acceso
6. HTTPS
7. LAMP

0 Introducción

Introducción

Antes de empezar con la instalación, configuración y administración de servidores web es bueno conocer las cuestiones fundamentales del protocolo HTTP. El protocolo de transferencia de hipertexto o HTTP (HyperText Transfer Protocol) establece el protocolo para el intercambio de documentos de hipertexto y multimedia en la web. El HTTP dispone de una variante cifrada mediante SSL llamada HTTPS. Teniendo en cuenta que HTTP es el protocolo utilizado en las comunicaciones web, conocer los aspectos básicos de este protocolo ayuda a entender algunas de las cuestiones que tienen que ver con la configuración y administración de servidores web.

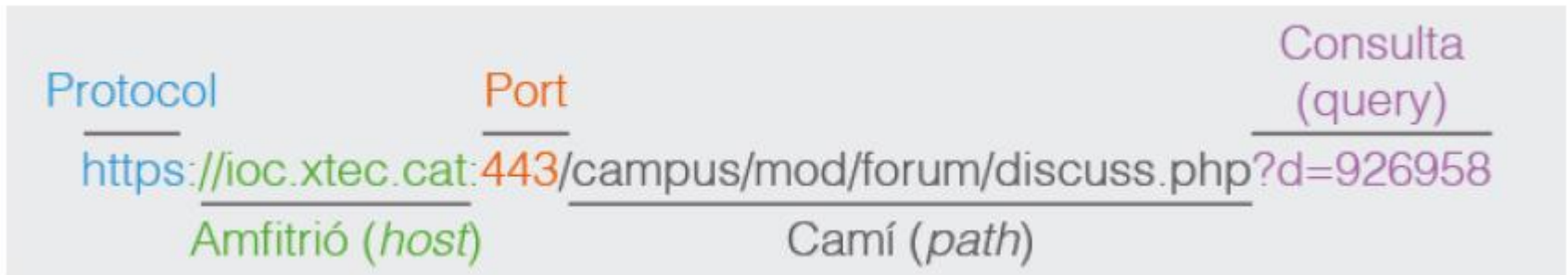
1 La URL

La URL

Los localizadores uniformes de recursos (URL, Uniform Resource Locator) son el mecanismo que permite, como indica su nombre, localizar recursos en internet mediante los distintos protocolos disponibles. Existe también el concepto de identificador uniforme de recursos (URI, Uniform Resource Identifier) que, en la práctica, se considera equivalente al concepto de URL, ya que un URI equivale al conjunto de un URL más un nombre uniforme de recursos (URN, Uniform Resource Name). Como que el concepto URN en la práctica no se usa, los términos URL y URI son equivalentes e intercambiables en la mayor parte de los casos.

La URL

Una URL está formada por varios elementos cada uno de los cuales nos describe un aspecto distinto sobre la localización de un recurso.



La URL

- Esquema (scheme): indica el protocolo que se utilizará para acceder al recurso especificado por el resto de la URL. Según el protocolo indicado (HTTP y HTTPS), el resto de la URL puede tener diferencias en su estructura.
- Información de usuario (user info): este elemento no forma parte de la definición estándar de los URL, pero muchos servidores web contemplan la opción de juntar información de usuario (nombre, y opcionalmente password) para facilitar procesos básicos de identificación y validación de usuarios.
- Alojador (host): identifica el servidor web. Puede ser un nombre de dominio o una dirección IP.

La URL

- Puerto: se trata de un elemento opcional que sirve para indicar qué puerto TCP/IP se utilizará para establecer la conexión para acceder al recurso. Con protocolo HTTP, si no se indica, toma por defecto el puerto 80 y con HTTPS se utiliza por defecto el puerto 443.
- Camino (path): indica la localización del recurso dentro del servidor. El camino asignado en una URL no tiene por qué representar un camino físico de disco con el mismo nombre o secuencia de directorios ya que los servidores web permiten hacer un mapeo entre caminos de URL y directorios de disco (directorios virtuales).

La URL

- Consulta (query): permite pasar parámetros adicionales útiles especialmente cuando el recurso al que accedemos es un script u otro tipo de elemento que ejecuta código en el servidor. Está formado por pares “nombre=valor”, los cuales, en caso de haber más de uno, se separan con el carácter &.
- Fragmento: permite especificar una sección específica dentro del recurso identificar por la URL. Los navegadores no envían esta parte de la URL a los servidores web. El navegador, una vez recibe la respuesta del servidor, si se ha especificado un fragmento, intenta localizar la sección identificada por el fragmento dentro del contenido entero que ha recibido y normalmente le focaliza en la visualización al usuario.

La URL

Muchos de los elementos de una URL son opcionales y se pueden omitir si el contexto donde se utiliza la URL permite asumir unos valores por defecto. Un ejemplo está en la barra de direcciones de un navegador, donde no es necesario escribir “http:’/’/” al inicio de la URL, ya que es el valor predeterminado para el navegador.

También pasa en URL para hacer referencia a recursos o enlaces dentro de un documento HTML. En este caso, además, se puede omitir el nombre de host e incluso una parte del camino de las URLs internas del documento HTML.

La URL

Podemos diferenciar entre:

URL absoluta

URL que incluye el nombre de alojador (host) y el camino (path) completo del recurso. Algunos ejemplos de URL absolutos son:

https://es.wikipedia.org/wiki/Localizador_uniforme_de_recursos
<https://ioc.xtec.cat/educacio/>

La URL

URL relativo

URL que no incluye el esquema (scheme), el host y el puerto. Cuando una URL es relativa se suele llamar camino (path) y se puede dividir en caminos completos y caminos relativos.

- Caminos completos (full paths): comienzan siempre con el carácter / y especifican toda la secuencia de directorios a recorrer hasta llegar al recurso dentro del mismo servidor del que se ha descargado el documento HTML, partiendo del directorio raíz del sitio web. Por ejemplo, /index.html.

La URL

- Caminos relativos (relative paths): comienzan con un carácter diferente de / e indican la secuencia de directorios a recurrir para llegar al recurso dentro del mismo servidor del que se ha descargado el documento HTML, partiendo del directorio del que se ha descargado el documento. Por ejemplo, ../ioc/images/hdr_bg.gif.

Los caracteres que se pueden utilizar sin restricciones en las URL son:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
0	1	2	3	4	5	6	7	8	9	-	_	.	~												

La URL

También se pueden usar otros caracteres, pero deben estar codificados para poder ser interpretados. Para ello, se usa la codificación porcentual (percent-encoding o URL encoding):

!	#	\$	&	'	()	*	+	,	/	:	;	=	?	@	[]
%21	%23	%24	%26	%27	%28	%29	%2A	%2B	%2C	%2F	%3A	%3B	%3D	%3F	%40	%5B	%5D

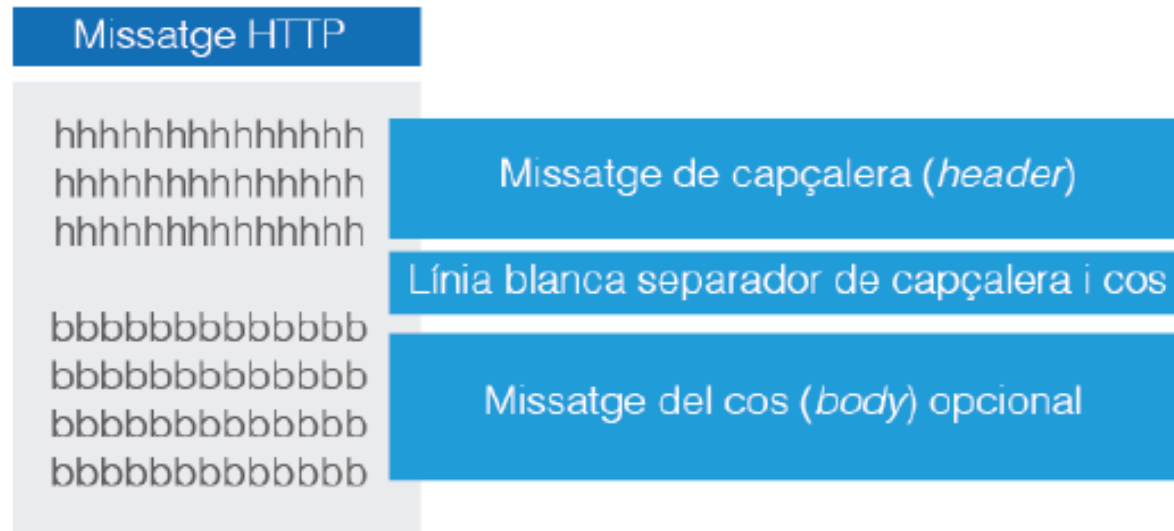
La URL

HTTP es un protocolo de petición/respuesta (request-response). El ciclo de comunicación entre el cliente y el servidor se basa en dos pasos donde la respuesta por parte del servidor viene precedida de una petición por parte del cliente.

Aunque las peticiones y respuestas contienen información diferente, ambas tienen una misma estructura formada por una cabecera y un cuerpo. La cabecera contiene información sobre el mensaje (metadatos) y el cuerpo es el que lleva el contenido del mensaje. El contenido de las peticiones y respuestas puede estar vacío en algunas ocasiones, por tanto, puede haber peticiones o respuestas sin contenido, sólo con cabecera.

La URL

Los mensajes HTTP, sean peticiones o respuestas, tienen una estructura formada por una cabecera y un cuerpo. La cabecera está formada por información alfanumérica y está separada del cuerpo por una línea en blanco. El contenido del cuerpo (que puede ser opcional) es alfanumérico o binario.



La URL

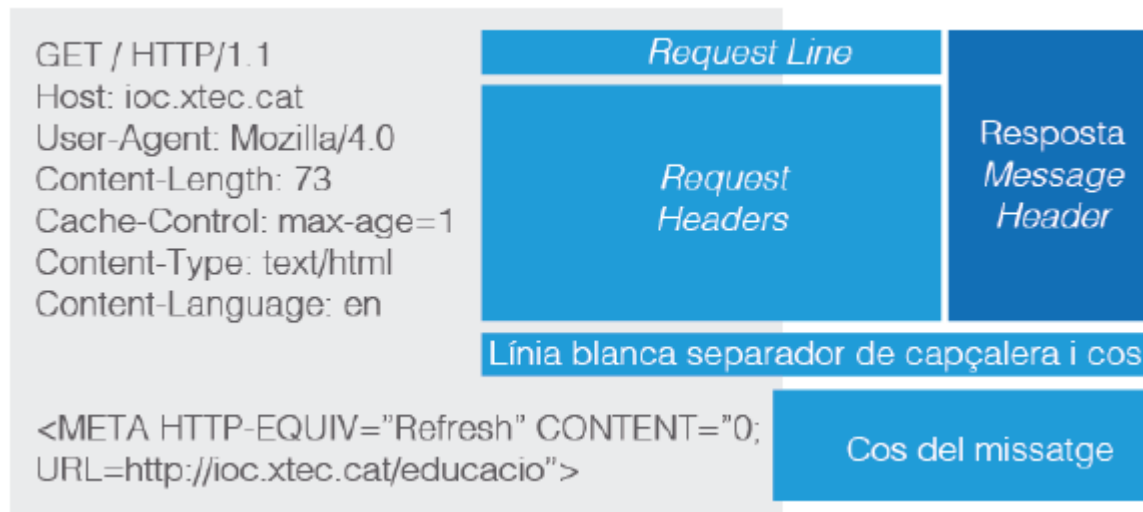
Las peticiones HTTP, en la primera línea de la cabecera, contienen la línea de petición (request line).

El resto de líneas (opcionales) contienen las cabeceras de petición (request headers).

- Línea de petición: primera línea de la cabecera de una petición. Está formada por tres campos:
 - Método de petición (request method): puede ser cualquiera de los métodos de petición que define el protocolo HTTP.
 - URL de petición (request URL): es la URL del recurso que se solicita.
 - Versión HTTP: es la versión de protocolo HTTP que se utilizará (HTTP/1.0 o HTTP/1.1).

La URL

- Cabecera de petición: cada cabecera va en una línea diferente y está formada por un par "nombre: valor". Si hay varios valores, éstos se separan con comas.



La URL

El protocolo HTTP tiene definidos 8 métodos de petición (request methods), que son el primero que se especifica en la línea de las peticiones.

1. GET: el método GET se utiliza para realizar la petición de un recurso o documento al servidor (un documento HTML, una imagen...), especificado en el campo URL de la línea de petición (request line). Las peticiones realizadas con el método GET sólo deberían hacer que el servidor devuelva algún tipo de respuesta sin que esto provoque la modificación de los datos del servidor ni de la aplicación web sobre la que se ha ejecutado con GET.

La URL

2. HEAD: el método HEAD se usa para simular una petición de un recurso al servidor. Funciona exactamente igual que el GET, con la diferencia de que el servidor sólo responde con las cabeceras de la respuesta (no envía el contenido). Se utiliza para poder realizar un pronóstico de respuesta de una hipotética petición GET. Es utilizado habitualmente por los navegadores para comprobar las cabeceras de un recurso determinado y así decidir si es necesario descargarlo o no en caso de tener una copia en la memoria cache local (las cabeceras nos pueden decir el tamaño del documento, la fecha de última modificación, etc.).

La URL

3. POST: es el método habitual para enviar los datos de formularios HTML que pueden provocar modificaciones en los datos del servidor o de la aplicación web. Puede parecer igual que el método GET, pero tiene algunas diferencias importantes:

1) Los parámetros de un GET van concatenados en la URL de la petición y es visible en la barra de direcciones del navegador. En cambio con POST se adjuntan como contenido del cuerpo de la petición y, por tanto, no son tanto visibles. Como consecuencia, las peticiones GET se pueden memorizar en los marcadores del navegador y pueden representarse, también, como URL en un enlace dentro de un documento HTML. Con POST, esto no es posible.

La URL

2) Las peticiones GET se consideran idempotentes (así lo define HTTP), lo que significa que deben poder ejecutarse tantas veces como se quiera sin que se produzcan modificaciones del estado o los datos del servidor. Por tanto, los navegadores permiten la ejecución repetida de peticiones GET. Por ejemplo, actualizando una página cargada con GET o echando atrás a una página previamente cargada con GET sin dar ningún aviso al usuario. En cambio, con el POST está al revés. Las peticiones POST se consideran no idempotentes (puede provocar modificaciones) y los navegadores, cada vez que se pretende repetir una petición POST, informan al usuario con un mensaje de confirmación.

La URL

4. PUT: el método PUT se utiliza para poder crear o reemplazar un determinado recurso o documento del servidor. Con el método PUT podemos solicitar que el contenido presente en el cuerpo de la petición se almacene en el servidor y pase a estar accesible a través de una URL que indicamos en la línea de petición. Si ya existe, se sustituiría el actual por lo que damos con el PUT. Si no existe, se crearía uno nuevo. La utilización de este método está restringido en la configuración por defecto de los servidores web ya que permitiría la modificación no autorizada de sus contenidos (Si lo probamos, podemos recibir un error del tipo "405 Method Not Allowed").

La URL

5. DELETE: el método DELETE se utiliza para poder eliminar un determinado recurso o documento del servidor. Con el método DELETE se puede solicitar en el servidor que elimine el archivo asociado a una URL del recurso indicado en la línea de petición. La utilización de este método está restringida en la configuración por defecto de los servidores web ya que permitiría la modificación no autorizada de sus contenidos (si lo probamos, podemos recibir un error del tipo “405 Method Not Allowed”).

La URL

6. CONNECT: este método se utiliza para poder pasar conexiones seguras SSL a través de conexiones HTTP y para poder gestionar conexiones HTTP a través de servidores intermediarios (proxies).

7. OPTIONS: el método OPTIONS pide al servidor qué métodos HTTP se pueden utilizar sobre el recurso identificado con una URL de la línea de petición.

8. TRACE: el método TRACE pide al servidor que devuelva una copia de las cabeceras de la petición tal y como las ha recibido. Este método HTTP suele estar desactivado por defecto. Una vez especificada la línea de petición de una petición HTTP, puede haber una serie de cabeceras de petición (opcionales) cada una de las cuales ocupa una línea y tiene el formato “nombre: valor”.

La URL

Las cabeceras más habituales en las peticiones HTTP enviadas a los navegadores son:

- Alojador (host): cabecera obligatoria en HTTP/1.1. El cliente debe especificar el nombre del host al que envía la petición (una misma máquina puede tener varios nombres de dominio asociados). Un mismo servidor con una única IP puede servir varios sitios/aplicaciones web con nombres de dominio distintos. Para poder identificar en cuál de ellos queremos enviar la petición usamos esta cabecera.

La URL

- Usuario-agente (user-agent): indica qué software cliente (habitualmente un navegador) utiliza el usuario. Suele ser una cadena que incluye un nombre identificador del navegador, un descriptor de versión y un descriptor de sistema operativo y plataforma sobre la que se ejecuta el navegador. Por ejemplo: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/29.0.1547.76 Safari/537.36

La URL

- **Accept:** el cliente puede utilizar esta cabecera para indicar al servidor los tipos MIME que es capaz de manejar y cuál es su orden de preferencia. Si el servidor tiene varias versiones del recurso solicitado puede consultar esta cabecera para decidir cuál se enviará al cliente. Este proceso se llama negociación de tipos de contenidos (content-type negotiation). Por ejemplo:
Accept: image/png,image/*;q=0.8,*/*;q=0.5

La URL

- Accept-Language: el cliente puede utilizar esta cabecera para indicar al servidor los idiomas preferidos para las respuestas obtenidas. Si el servidor tiene el recurso solicitado en varios idiomas, puede consultar esta cabecera por decidir qué versión enviará al cliente. Este proceso se llama negociación de idioma (language negotiation). Por ejemplo: Accept-Language: es,es
- Accept-Charset: el cliente puede utilizar esta cabecera para indicar al servidor el juego de caracteres preferido para las respuestas obtenidas que incluyan información alfanumérica. Este proceso se llama negociación de juego de caracteres (charset negotiation). Por ejemplo: Accept-charset: utf- 8, iso8859-1

La URL

- Content-Type y content-Length: cuando la petición tiene un cuerpo, es decir, uno contenido, en la cabecera debe especificarse el tipo (con un identificador MIME) y el tamaño (en bytes) mediante estas dos cabeceras.
- Referer: esta cabecera permite al cliente especificar en el servidor la dirección (URL) del recurso de donde se ha obtenido la URL de la petición que se le está enviando.
- Accept-Encoding: el cliente puede utilizar esta cabecera para indicar al servidor los tipos de codificación que soporta. Si el servidor tiene el recurso comprimido o es capaz de comprimirlo al vuelo en alguno de los formatos que acepta el cliente, puede utilizarlo para reducir el tiempo de transmisión. El servidor debe indicar en la cabecera Content-Encoding de la respuesta el tipos de compresión que ha aplicado.

La URL

- Connection: el cliente puede utilizar esta cabecera para indicar al servidor si tiene que cerrar la conexión una vez enviada la respuesta a la petición recibida, o dejar la conexión abierta a la espera de recibir nuevas peticiones sin tener que repetir el proceso de establecimiento de la conexión. Puede tener dos valores: close o keep-alive.

La URL

Las respuestas HTTP, en la primera línea de la cabecera, contienen la línea de estatus. El resto de líneas (opcionales) contienen las cabeceras de respuesta.

- Línea de estatus (status line): primera línea de la cabecera de una respuesta. Está formada por tres campos:
 - Versión HTTP: versión de protocolo HTTP que utiliza el servidor (HTTP/1.0 o HTTP/1.1).
 - Código de estatus: código de tres dígitos que indica el resultado de la petición.
 - Reason phrase: pequeña explicación del significado del código de estatus.

La URL

El protocolo HTTP define 5 códigos de estado (status codes) que permiten indicar varios tipos de situaciones que se pueden dar como respuesta a una petición de un cliente.

- 1xx: son de tipo informativo, informan al cliente que la petición ha sido recibida y que el servidor sigue procesando la respuesta.
- 2xx: indican la petición ha sido correcta y se ha procesado satisfactoriamente.

La URL

- 3xx: indican alguna forma de redirección. Con un código de esta serie se da a entender al cliente que la petición es correcta pero que la respuesta se ha de obtener de algún otro sitio.
- 4xx: indican que ha habido un error en el procesamiento de la petición porque el cliente ha hecho algo mal, el error ha sido causado por cliente.
- 5xx: indican que ha habido un error en el procesamiento de la petición debido a un fallo en el servidor, el error ha sido causado por el servidor.

La URL

Una vez especificada la línea de estado de una respuesta HTTP, hay una serie de cabeceras de respuesta (opcionales) cada una de las cuales ocupa una línea y tiene el formato “nombre: valor”. Las cabeceras más habituales en las respuestas HTTP son:

- Content-Base: cabecera que contiene una URL para ser utilizado como base para las URL relativas que haya en los documentos HTML que se envíen con la respuesta.
- Content-Length: cabecera de respuesta que indica el tamaño en bytes del cuerpo asociado a la respuesta.

La URL

- Content-Type: cabecera que indica el tipo de contenido del cuerpo asociado a la respuesta. Los media types reconocidos están definidos por la Internet Assigned Numbers Authority (IANA).
- Date: esta cabecera, obligatoria en todas las respuestas en HTTP/1.1, indica la fecha y hora de envío de la respuesta, es decir, cuando sale del servidor.
- Last-Modified: cabecera que indica la fecha y hora en la que el recurso pedido ha sido actualizado por última vez. El objetivo de esta cabecera es permitir la gestión de caché de recursos. No funciona correctamente para recursos dinámicos. Para solucionarlo se introdujo la cabecera ETag.

La URL

- ETag: permite que el servidor pueda enviar un hash o checksum del cuerpo de la respuesta etiqueta de entidad (entity tag) para ser utilizado para el control de caché de recursos HTTP en los clientes y servidores proxies. Se puede considerar que todas las copias de un recurso con la misma URL y la misma etiqueta de entidad son idénticos. Por tanto, si se tiene una copia en memoria cache no es necesario descargarlos de nuevo.
- Server: es el equivalente a la cabecera de petición User-Agent, pero de al lado del servidor. Sirve para que el servidor web pueda indicar su nombre y versión.
- Location: indica al cliente hacia dónde debe ir a buscar un recurso que haya pedido y que no se encuentra en la URL de la petición (respuesta con un código de estado de la serie 3xx).

2 Configuración de Apache

Configuración de Apache

El conjunto de instrucciones de instalación o configuración mediante órdenes de sistema es dependiente de éste (si va a un sistema REDHAT, CENTOS, etc. debe mirar el procedimiento de instalación y las órdenes específicas del sistema).

La configuración interna es independiente del sistema operativo que utilizamos, cualquier configuración que realice durante estos apartados es compatible con otros sistemas operativos basados en Unix.

Configuración de Apache

Para iniciar el proceso de instalación de Apache, primero abra un terminal y ejecute los comandos de actualización de los repositorios de Ubuntu:

```
1 sudo apt-get update
```

```
1 sudo apt-get install apache2
```

Configuración de Apache

Una vez finalizada la instalación proceda a verificar el resultado de ejecutar el siguiente orden:

```
1 sudo service apache2 status
```

```
daw@daw:~$ sudo service apache2 status
[sudo] password for daw:
• apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2022-10-05 14:39:10 UTC; 6 days ago
     Docs: https://httpd.apache.org/docs/2.4/
  Process: 646 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
  Process: 4810 ExecReload=/usr/sbin/apachectl graceful (code=exited, status=0/SUCCESS)
 Main PID: 710 (apache2)
    Tasks: 55 (limit: 2238)
   Memory: 7.5M
      CPU: 15.123s
   CGroup: /system.slice/apache2.service
           └─ 710 /usr/sbin/apache2 -k start
              └─ 4816 /usr/sbin/apache2 -k start
                 └─ 4817 /usr/sbin/apache2 -k start
```

Configuración de Apache

Si todo ha ido bien debe mostrar una página web para informar de la correcta instalación del servidor Apache



This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuración de Apache

Una vez verificado que Apache funciona, la configuración del servidor se realiza mediante un archivo de texto plano, al igual que se suele hacer con muchos servicios de Unix, como por ejemplo vsftpd.

El directorio donde están las configuraciones está dentro de `/etc/apache2`, y el archivo de configuración principal es el archivo `apache2.conf`.

Configuración de Apache

Editar el archivo original puede comportar modificar el archivo y perder los datos iniciales de configuración. Siempre es bueno hacer una copia de seguridad del archivo con el mandato cp (para copiar archivos).

Navigate hasta el directorio de configuración de Apache /etc/apache2 y ejecute:

```
1 ls -l
```

Configuración de Apache

Con este comando se lista todo el contenido del directorio, con el siguiente resultado:

```
daw@daw:/$ cd /etc/apache2/  
daw@daw:/etc/apache2$ ls -l  
total 80  
-rw-r--r-- 1 root root 7224 jun 14 12:30 apache2.conf  
drwxr-xr-x 2 root root 4096 sep 30 16:17 conf-available  
drwxr-xr-x 2 root root 4096 sep 30 16:17 conf-enabled  
-rw-r--r-- 1 root root 1782 mar 23 2022 envvars  
-rw-r--r-- 1 root root 31063 mar 23 2022 magic  
drwxr-xr-x 2 root root 12288 sep 30 16:17 mods-available  
drwxr-xr-x 2 root root 4096 sep 30 16:17 mods-enabled  
-rw-r--r-- 1 root root 320 mar 23 2022 ports.conf  
drwxr-xr-x 2 root root 4096 sep 30 16:17 sites-available  
drwxr-xr-x 2 root root 4096 sep 30 16:17 sites-enabled
```

Configuración de Apache

Como puede ver, el directorio `/etc/apache` contiene diferentes archivos, incluyendo `apache2.conf`. Los archivos existentes hacen referencia a archivos de configuración de modularidades que amplían la capacidad de Apache, como puede ser: añadir módulos como PHP, configurar la seguridad para cifrar la información, crear servidores virtuales, etc.

Inicie la edición del archivo `apache2.conf`.

Dentro del directorio haga:

```
1  sudo nano apache2.conf
```

Configuración de Apache

Como puede comprobar, el archivo de configuración `apache2.conf` es bastante extenso. A continuación se detallan las opciones de configuración más relevantes. Para verificar sintaxis y configuraciones le recomendamos que visite el anexo donde se explican las sintaxis y las configuraciones permitidas.

Dentro de la configuración inicial, se pueden encontrar algunas de las directivas más importantes:

- Timeout: número de segundos antes de que reciba y envía el tiempo de espera.
- Keep-alive: permite aceptar conexiones persistentes.

Configuración de Apache

- `MaxKeepAliveRequests`: número máximo de solicitudes de permiso durante una conexión persistente.
- `KeepAliveTimeout`: número de segundos de espera para la siguiente petición del mismo cliente en la misma conexión.
- `ErrorLog`: ubicación del archivo de registro de errores.
- `Include module configuration`: camino donde se encuentra el archivo de módulos disponibles y activos.
- `Include list of ports to listen`: camino donde se encuentra el archivo de los puertos de escucha del servidor.

Configuración de Apache

- DocumentRoot: establece el directorio de publicación de su web principal o por defecto.
- Directory: para cada directorio que sea necesario configurar se puede definir un block de opciones de configuración agrupadas en esta directiva.
- DirectoryIndex: documentos que se muestran por defecto cuando se solicita un URL y no se especifica el documento.
- AccessFileName: nombre del archivo que debe buscar en cada directorio para directivas de configuración adicionales.

Configuración de Apache

- Include the virtual host configurations: camino donde se encuentra el archivo de configuración de los servidores virtuales.
- User: cuenta de usuario que el servidor web Apache utilizará mientras se ejecute (determina los permisos de acceso que tendrá sobre directorios y archivos).
- Group: grupo de usuarios que el servidor web Apache utilizará mientras se ejecute (al igual que con el usuario, determina los permisos de acceso que tendrá sobre directorios y archivos).

Configuración de Apache

Con las opciones de instalación por defecto, ya tendrá el sitio web por defecto en la carpeta:

```
1 /var/www
```

Este sitio web, por defecto, es el ofrecido para cualquier conexión que se realice sobre el puerto 80 en cualquiera de las direcciones IP o nombres de dominio establecidos sobre el servidor. Para modificar el puerto es necesario modificar el archivo `ports.conf`.

3 Módulos: instalación, configuración y uso

Módulos: instalación, configuración y uso

La gran mayoría de servidores web permiten la instalación de módulos para ampliar sus funcionalidades. Tener funcionalidades en forma de módulo permite adaptarse mejor el consumo de recursos del servidor web a nuestras necesidades de producción (el servidor web sólo cargará y ejecutará los módulos que como administradores tenemos instalados y configurados).

El servidor web Apache HTTP Server permite añadir funcionalidades adicionales a las que nos ofrece su configuración básica en forma de módulos instalables.

Módulos: instalación, configuración y uso

Instalación de módulos en Apache HTTP Server

Trabajando en una plataforma con un sistema Debian, la instalación de módulos se realiza normalmente mediante el gestor de instalación de paquetes APT. La instalación de un módulo añade los siguientes archivos en el sistema:

- El archivo del módulo (habitualmente el nombre suele empezar por mod y terminar en .so) se guarda en el directorio `/usr/lib/apache2/modules`. Si hace un `ls -l` de `/usr/lib/apache2/modules` puede ver el contenido de la carpeta similar a:

Módulos: instalación, configuración y uso

Instalación de módulos en Apache HTTP Server

```
1 ioc@ioc:/usr/lib/# ls -l
2 total 2732
3 -rw-r--r-- 1 root root 13937 jul 15 2016 httpd.exp
4 -rw-r--r-- 1 root root 10256 jul 15 2016 mod_access_compat.so
5 -rw-r--r-- 1 root root 10248 jul 15 2016 mod_actions.so
6 ...
7 -rw-r--r-- 1 root root 14352 jul 15 2016 mod_usertrack.so
8 -rw-r--r-- 1 root root 10256 jul 15 2016 mod_vhost_alias.so
9 -rw-r--r-- 1 root root 22536 jul 15 2016 mod_xml2enc.so
```


Módulos: instalación, configuración y uso

Instalación de módulos en Apache HTTP Server

- El archivo que contiene la directiva para cargar del módulo (LoadModule) tiene como a extensión .load, y se guarda en el directorio /etc/apache2/mods-available.
- El archivo que contiene las directivas de configuración del módulo (en caso de que en tenga), con una configuración por defecto, suele tener una extensión .conf, y está almacenado en el directorio /etc/apache2/mods-available. No sin embargo, no todos los módulos vienen con el correspondiente archivo de directivas de configuración.

Módulos: instalación, configuración y uso

Activación y desactivación de módulos en Apache HTTP Server

Una vez instalados los módulos, deben activarse. Con servidores Apache se pueden tener módulos instalados pero no activados, es decir, que no se están utilizando.

Para activar un módulo es necesario crear un enlace simbólico dentro del directorio `/etc/apache2/mods-enabled` sobre el archivo `.load` y sobre el archivo `.conf` (si tiene archivo asociado) que contiene la directiva de carga del módulo y las directivas de configuración respectivamente. Estos archivos son `/etc/apache2/mods-available`. Para facilitar la configuración y la administración del servidor suele ponerse el mismo nombre del archivo en el enlace simbólico.

Módulos: instalación, configuración y uso

Activación y desactivación de módulos en Apache HTTP Server

También se pueden activar los módulos usando las herramientas que facilitan el servidor Apache y el sistema operativo. Dispone del comando `a2enmod`, que permite activar un módulo siempre que esté instalado. Si le da como parámetro el nombre del módulo que se debe activar, crea el enlace o enlaces simbólicos necesarios en el directorio `/etc/apache2/mods-enabled`.

Por ejemplo:

```
1 a2enmod rewrite
```

Módulos: instalación, configuración y uso

Activación y desactivación de módulos en Apache HTTP Server

Una vez instalados y activados los módulos, ya pueden utilizarse. A partir de ahí, puede cambiar la configuración manipulando el archivo `.conf` que esté asociado al módulo. Si no existe este archivo, puede que el módulo no tenga directivas de configuración o porque por defecto no lleva el archivo (existe la posibilidad de que se pueda crear y poner algunas directivas). Para poder cambiar la configuración de un módulo, debe consultar la documentación técnica para saber qué directivas de configuración ofrece y qué opciones posibles existen para cada directiva.

Módulos: instalación, configuración y uso

Activación y desactivación de módulos en Apache HTTP Server

Para desactivar un módulo existen dos opciones, igual que antes. O bien borre los enlaces simbólicos que ha creado o utilice el mandato `a2dismod`.

Por ejemplo:

```
1 a2dismod rewrite
```

Módulos: instalación, configuración y uso

Activación y desactivación de módulos en Apache HTTP Server

Los módulos de Apache son opcionales (no tienen por qué estar cargados), sus directivas de configuración de un módulo en concreto pueden provocar errores en caso de que el módulo no esté activado. Para evitar esto, Apache dispone del bloque `<IfModule>`, que se puede incluir en sus archivos de configuración que permiten indicar que unas determinadas directivas de configuración surtan efecto sobre el servidor sólo si el módulo que se indica está activo.

```
1 <IfModule nom_mòdul.c>
2   # Directives de configuració del mòdul, en cas que estigui activat.
3   ...
4 </IfModule>
```

Módulos: instalación, configuración y uso

Activación y desactivación de módulos en Apache HTTP Server

Para saber qué módulos están activos, puede listar el contenido del directorio con `/etc/apache2/mods-enabled`.

Para saber qué módulos hay disponibles para instalar en el servidor, con el gestor de paquetes APT puede ejecutar el comando `apt-cache search libapache2-mod`. Se mostrará un listado con el nombre del módulo y una pequeña descripción de cada uno.

```
1 ioc@ioc:/etc/apache2# ls /etc/apache2/mods-enabled/*.load
2 /etc/apache2/mods-enabled/access_compat.load
3 /etc/apache2/mods-enabled/alias.load
4 /etc/apache2/mods-enabled/auth_basic.load
5 /etc/apache2/mods-enabled/authn_core.load
6 /etc/apache2/mods-enabled/authn_file.load
7 /etc/apache2/mods-enabled/authz_core.load
8 /etc/apache2/mods-enabled/authz_host.load
9 /etc/apache2/mods-enabled/authz_user.load
10 /etc/apache2/mods-enabled/autoindex.load
11 /etc/apache2/mods-enabled/deflate.load
12 /etc/apache2/mods-enabled/dir.load
13 /etc/apache2/mods-enabled/env.load
14 /etc/apache2/mods-enabled/filter.load
15 /etc/apache2/mods-enabled/mime.load
16 /etc/apache2/mods-enabled/mpm_event.load
17 /etc/apache2/mods-enabled/negotiation.load
18 /etc/apache2/mods-enabled/setenvif.load
19 /etc/apache2/mods-enabled/status.load
```

4 Servidores virtuales. Creación, configuración y utilización

Servidores virtuales

El protocolo DNS permite tener varios nombres de dominio que apuntan sobre un mismo servidor (puede ser una misma IP o pueden ser direcciones IP distintas que correspondan a un mismo servidor).

Mediante el protocolo DNS se puede conseguir que un servidor ofrezca sitios webs diferentes en función del nombre de dominio que se haya utilizado para establecer la conexión, gracias también a la cabecera host que proporciona el protocolo HTTP. Esto es lo que se conoce como servidores virtuales (un mismo servidor que actúa como si fueran varios servidores web).

Servidores virtuales

En la terminología de Apache se llama virtual host o vhost a cada uno de los servidores virtuales que están en funcionamiento aparte del servidor principal o por defecto:

- Cuando se asignan servidores virtuales distintos a direcciones IP diferentes se habla de servidores virtuales basados en IP o IP-based vhosts.
- Cuando se asignan múltiples sedes virtuales a una misma dirección IP se habla de servidores virtuales basados en nombre o Name-based vhosts.

Servidores virtuales

Los pasos a seguir para crear y poner en marcha un servidor virtual son:

1. Vaya al directorio `/etc/apache2/sites-available`. Esta carpeta contiene los archivos de configuración para cada uno de los servidores virtuales disponibles en servidor.
2. Cree un nuevo archivo de configuración del sitio web con el nombre que desee, dentro del mismo directorio.
3. Añada las opciones para adaptarlo a las necesidades del nuevo servidor virtual dentro de este archivo.

Servidores virtuales

4. Cree el enlace simbólico dentro de la carpeta `/etc/apache2/sites-enabled` que apunte hacia el archivo de configuración del servidor virtual creado en la carpeta `/etc/apache2/sites-available`. O use el comando `a2ensite nombre_archivo_servidor_virtual.conf`.
5. Ejecute el mandato `service apache2 reload`. Este comando fuerza al servidor Apache a volver a cargar su configuración.
6. Compruebe cómo responde el servidor facilitando el sitio web correspondiente a cada servidor virtual.

Servidores virtuales

Vea un ejemplo de configuración de un servidor virtual:

```
1 <VirtualHost *:80>
2
3 # Aquest servidor virtual serà el que actuarà quan a la capçalera Host d'una
4 # petició HTTP hi trobem "www.iocdaw.cat".
5
6     ServerName www.iocdaw.cat
7
8 # Directori arrel del lloc web (directori físic de disc).
9
10    DocumentRoot /daw/m08
11
12 # Opcions de configuració per al directori arrel del lloc web.
13    <Directory /webs/enciams>
14        Options -FollowSymLinks -Indexes +MultiViews AllowOverride None
15        Order allow,deny
16        Allow from all
17    </Directory>
18
19 # Opcions de configuració dels logs del servidor virtual.
20    ErrorLog ${APACHE_LOG_DIR}/daw.error.log
21    LogLevel warn
22    CustomLog ${APACHE_LOG_DIR}/daw.access.log combined
23
24 </VirtualHost>
```

Servidores virtuales

Véase el análisis detallado de las principales opciones de configuración necesarias para definir una sede virtual:

- VirtualHost: esta directiva es la que hace el vínculo con la dirección IP y puerto asignados a la sede virtual. Aunque, por claridad, en el ejemplo se ha indicado un nombre de host (host) en lugar de una dirección IP, Apache recomienda usar siempre la dirección IP.
- ServerAdmin: indica el nombre del administrador de la sede virtual. De hecho, indica el correo electrónico.

Servidores virtuales

- DocumentRoot: define el directorio de publicación de su web virtual. El directorio que se indica es una ruta absoluta del sistema físico de archivos, no una ruta relativa del servidor web.
- ServerName: es el nombre virtual con el que se reconoce esta web, el nombre que los clientes deben referenciar para poder acceder a la web.
- errorLog y CustomLog: estas dos directivas especifican la ubicación de los archivos de registro o logs de monitorización de la actividad de esta sede web. Las rutas que se indican son relativas y se utiliza el directorio de logs definido en la configuración global.

Servidores virtuales

Hay que tener presente algunas cuestiones relativas a los permisos de acceso a carpetas y archivos:

En el directorio `/etc/apache2` hay un archivo llamado `envvars` donde, entre otros aspectos, se define el nombre de usuario y el grupo con el que trabajará el servidor Apache.

Por defecto encontraremos:

```
export APACHE_RUN_USER=www-data  
export APACHE_RUN_GROUP=www-data
```

Esto indica que el servidor Apache accederá a archivos y directorios utilizando el usuario `www-data` y el grupo `www-data`. Teniendo en cuenta este tema, será necesario dar permiso de lectura a los archivos (`r-`) y acceso a los directorios (`r-x`), como mínimo, a este usuario/grupo en todos los directorios y archivos que formen parte de los sitios web que sirva Apache.

5 Autenticación y control de acceso

Autenticación y control de acceso

Hasta ahora ha visto cómo crear y configurar varias sedes web accesibles para todos que tenga acceso al servidor. Hay ocasiones en las que se quiere restringir el acceso a una parte de la web o en toda la web pero sólo para usuarios concretos. En este apartado se describen diversas formas de hacerlo.

El servicio web incorpora mecanismos básicos para verificar a los usuarios que quieren acceder a áreas restringidas. Pero, además, la flexibilidad de los módulos hace que se puedan añadir nuevos mecanismos que puedan surgir más adelante aunque no hayan sido desarrollados por Apache. Así, para validar el acceso a un directorio con material de los profesores en una web de una escuela seguramente basta con el mecanismo básico de verificación de usuarios y grupos. En cambio, para acceder a un web ultrasecreto de una agencia gubernamental quizás es necesario incorporar mecanismos adicionales, basados por ejemplo en la huella óptica y el registro de voz.

Autenticación y control de acceso

Primeramente es necesario analizar los mecanismos de validación de usuarios generales que permite el servidor web:

- Autenticación básica con archivos: el mecanismo más simple para implementar el control de acceso a recursos de una sede web es utilizar archivos de usuarios y grupos propios del servidor. Apache proporciona herramientas para su creación. La principal ventaja de este método es la facilidad de administración. El inconveniente es que comporta una gestión diferenciada de los usuarios del servicio web y de los del sistema. De hecho, esto puede ser un inconveniente o una ventaja, si lo que interesa es tenerlos segregados.

Autenticación y control de acceso

- Autenticación mediante PAM: en los sistemas GNU/Linux actuales la autenticación de los usuarios se realiza vía PAM (Pluggable Authentication Module). El PAM comprueba el directorio `/etc/passwd`, LDAP, Kerberos, las huellas dactilares o lo que sea necesario. Usar el vínculo con el módulo del PAM es un buen mecanismo para validar a los usuarios del servicio web al igual que se validan los usuarios del sistema.
- Autenticación mediante LDAP: uno de los mecanismos más populares actualmente para la autenticación es el LDAP. El módulo del LDAP permite pasar la validación de los usuarios al encargado de gestionar la autenticación LDAP de los usuarios del sistema. También se puede tener en funcionamiento un servicio LDAP específico para las validaciones del servicio web.

Autenticación y control de acceso

Algunos conceptos clave relacionados con el control de acceso al servidor son:

- Autenticación: el proceso de autenticación es el que determina si un usuario es quien dice ser. En ningún momento gobierna qué derechos tiene, qué puede hacer y qué no, simplemente se encarga de comprobar que el usuario es quien dice ser. Para implementar la autenticación existen innumerables sistemas, desde los archivos de usuarios y contraseñas hasta sofisticados mecanismos de huellas dactilares, ópticas, datos biométricos o lápices USB (sin el lápiz el usuario no se puede identificar).

Autenticación y control de acceso

- Autorización: una vez identificado un usuario (es quien dice ser), ¿qué puede hacer?, ¿a qué recursos puede acceder?, ¿a qué no? Esto es la autorización: determinar los derechos de utilización de los recursos.
- Recurso con acceso restringido: el control de acceso al servidor busca determinar qué recursos son accesibles para los usuarios. Puede restringir el acceso a toda una sede web de modo que sólo los usuarios autorizados puedan acceder a sus contenidos. A menudo se restringen áreas concretas de su web, por ejemplo directorios que son accesibles sólo para un conjunto de usuarios (los empleados, o los profesores, en la web de la escuela). En este caso hablamos de directorios con acceso restringido.
- Reialme: en una sede web puede haber varias áreas restringidas a perfiles de usuario distintos. Los reinos permiten definir qué áreas restringidas comparten el mismo grado de acceso.

Autenticación y control de acceso

- Web con inicio de nombre de usuario/contraseña: un error muy típico es confundir la autenticación de servidor con la autenticación de software que hacen sus webs. Cuando un usuario se valida en un entorno web como por ejemplo Yahoo o Google, no está usando la autenticación con el servidor web. Está usando un usuario y una contraseña de la empresa web a la que se conecta y la gestión de esta sesión de usuario para consultar su correo se realiza mediante la programación en las mismas páginas web que visita. Esto nada tiene que ver con el control de acceso al servidor que se trata en este apartado.

Autenticación y control de acceso

Autenticar a los usuarios es determinar de forma veraz si un usuario es quien dice ser. Autorizar es indicar que usuarios tienen derecho a acceder a qué recursos. Sus web y directorios que limitan el acceso a un conjunto restringido de usuarios se llaman recursos restringidos. Los recursos restringidos que implementan la misma política de seguridad se pueden agrupar en reinos.

Autenticación y control de acceso

Módulos de control de acceso

Apache gestiona la autenticación y el control de acceso al servidor mediante módulos propios (y también se pueden incorporar módulos externos). Cada módulo consta de un conjunto de directivas que permiten configurar el funcionamiento de la autenticación y control de acceso implementados. Estos módulos se pueden clasificar en tres categorías según su funcionalidad:

1) Tipo de autenticación: la autenticación puede ser de tipo básico o digesto. En estos ejemplos se utiliza autenticación básica. La autenticación digesto implica comunicaciones cifradas. Estos módulos se implementan con la directiva `AuthType`.

```
AuthType Basic
```

Puede observar que los módulos identifican en su nombre la cadena `auth` de autenticación.

```
mod_auth_basic  
mod_auth_digest
```

Autenticación y control de acceso

Módulos de control de acceso

2) Proveedor de autenticación: indica cuál es el mecanismo usado para realizar la autenticación. Son los módulos que permiten autenticar usando archivos de contraseñas o el módulo PAM o el de LDAP. Se pueden identificar los módulos de esta familia porque incluyen en su nombre la cadena authn de authentication.

```
mod_authn_file  
mod_authn_alias  
mod_authnz_ldap  
...
```

Autenticación y control de acceso

Módulos de control de acceso

3) Autorización: los módulos de esta familia proporcionan autorización de usuario, de grupos, del LDAP o de lo que convenga. Estos módulos se determinan según el valor que tome la directiva `require`.

```
Require user -validuser
```

Puede observar que los módulos identifican en su nombre la cadena `authz` de authorization.

```
mod_authz_user  
mod_authz_group  
mod_authz_owner  
mod_authz_ldap  
...
```

Autenticación y control de acceso

Autenticación básica con ficheros

El mecanismo más sencillo para implementar la autenticación en el servidor es la autenticación básica con archivos de usuarios y grupos específicos para el servidor web. Esto puede interpretarse como una desventaja porque obliga a llevar una gestión de usuarios además de la gestión de usuarios del sistema. Pero al mismo tiempo es una ventaja si lo que queremos es segregar estos dos conjuntos de usuarios y administrarlos por separado.

Con la autenticación básica utilizando archivos se pueden validar los usuarios utilizando un archivo de usuarios, que contiene las cuentas de usuarios y sus contraseñas. También se pueden validar grupos de usuarios con un archivo de grupos, que indica qué usuarios forman parte de cada grupo.

Autenticación y control de acceso

Autenticación básica con ficheros

El proceso más simplificado para implementar la verificación de usuarios y grupos mediante archivos de texto plano con contraseñas requiere los siguientes pasos:

1. Crear el archivo de usuarios en el que se indica la contraseña correspondiente a cada usuario.
2. Crear el archivo de grupos asignando a cada grupo los usuarios que forman parte del mismo.
3. Identificar (o crear) el recurso que debe tener el acceso restringido.
4. Definir las directivas apropiadas para restringir el acceso al recurso a los usuarios autorizados.

Autenticación y control de acceso

Autenticación básica con ficheros

El siguiente ejemplo creará un directorio llamado m08 en la web www.iocdaw.cat, al que sólo podrán acceder los usuarios autorizados (los profesores).

Primero es necesario crear el archivo de usuarios. Se trata de un archivo de texto plano en el que se almacenan el identificador y la contraseña, que puede ser en texto plano o cifrada, de cada usuario. Para crear el archivo y cada nuevo usuario se utiliza el mandato `htpasswd`, proporcionada por el paquete del servidor. En el primer ejemplo se utiliza la opción `-c`, que crea el archivo de nuevo.

Autenticación y control de acceso

Autenticación básica con ficheros

```
1 ioc@ioc:/var/www# htpasswd -c passwd/passwd alumne1
2 New password:
3 Re-type new password:
4 Adding password for user alumne1
5 ioc@ioc:/var/www# htpasswd passwd/passwd alumne2
6 ioc@ioc:/var/www# htpasswd passwd/passwd professor
7 ioc@ioc:/var/www# cat passwd/passwd
8 alumne1:$apr1$pEDPMqo8$ITpyGspQph.EjTvHpZIm11
9 alumne2:$apr1$zeHdfxng$iKxwbzVAGItP/2Fq8BniM.
10 professor:$apr1$Uje7o6eZ$B7c5ACGE8WH2bdZqzc0kz1
```

Autenticación y control de acceso

Autenticación básica con ficheros

A continuación hay que poner en cada grupo (de momento no hay ninguno) los usuarios que deben formar parte. De hecho, es tan sencillo como crear un archivo de texto plano cada línea del que consta del nombre del grupo, el delimitador dos puntos (:) y la lista de usuarios separados por espacios. Puede observar que la usuaria anna está en ambos grupos.

```
1 ioc@ioc:/var/www# cat passwd/group
2 alumnes: alumne1 alumne2
3 profes: professor
```


Autenticación y control de acceso

Autenticación básica con ficheros

Ahora es necesario generar el recurso restringido, cuyo acceso sólo se permitirá a los usuarios autorizados. En este caso será un directorio llamado m08 en la web www.iocdaw.cat.

```
1 ioc@ioc:/var/www# mkdir m08
2 ioc@ioc:/var/www# nano m08/index.html
3 ... crear una pàgina ....
```

Autenticación y control de acceso

Autenticación básica con ficheros

Finalmente debe asignarse al directorio local las directivas apropiadas para convertirlo en un recurso de acceso restringido. Habrá que modificar el archivo de configuración global `httpd.conf` y definir un `blog` de configuración usando la directiva `Directory`. En esta directiva debe indicarse la ruta absoluta correspondiente al sistema de archivos real del servidor (no es posible utilizar rutas relativas al servicio web).

```
1 <Directory --pathabsolutfilesystem>  
2 ... opciones de configuració ...  
3 </Directory>
```

Autenticación y control de acceso

Autenticación básica con ficheros

Un ejemplo completo de configuración es el que se muestra a continuación, en el que únicamente se permite acceder al recurso a usuarios del grupo de los profesores llamado profes:

```
1 <Directory /var/www/m08>
2   AuthType Basic
3   AuthName "Restringit a professors"
4   AuthBasicProvider file
5   AuthUserFile /var/www/passwd/passwd
6   AuthGroupFile /var/www/passwd/group
7   Require group profes
8 </Directory>
```

Autenticación y control de acceso

Autenticación básica con ficheros

Repasamos las directivas que se utilizan:

AuthType: indica que el tipo de autenticación es básica (en lugar de digesto).

AythName: declara el reino al que pertenece el recurso restringido. Esto permite que si existen otros recursos restringidos asociados a este reino el usuario que ya se ha autenticado en uno de ellos no deba hacerlo en los demás. El nombre del reino lo pone el administrador web.

AuthBasicProvider: indica el método de autenticación a usar. Puede tomar valores tipo ldap, pam, dbm, bdb, file y otros. El valor file significa que se utilizará un archivo de usuarios y opcionalmente uno de grupos.

Autenticación y control de acceso

Autenticación básica con ficheros

AuthUserFile: indica cuál es el archivo que contiene las cuentas de los usuarios locales del servidor Apache. Es el archivo creado en el ejemplo anterior.

AuthGroupFile: indica cuál es el archivo de grupos en el que consta qué grupos de usuarios existen y qué usuarios pertenecen a cada grupo.

Require: esta directiva es la que determina cuál es la autorización que se debe hacer. En el ejemplo se permite que cualquier usuario del grupo `profes` tenga acceso al recurso.

Autenticación y control de acceso

Autenticación básica con ficheros

Por último, es necesario verificar que el acceso al directorio local es concedido únicamente a los miembros del grupo profes.

Evidentemente el mecanismo más sencillo es verificar desde de un navegador el acceso al recurso www.iocdaw.cat/m08 y observar que se pide la autenticación.

Autenticación y control de acceso

Ejemplos de mecanismos de autorización

La directiva require es la que define la autorización de acceso al recurso, es decir, quién puede acceder. Estos son algunos ejemplos de uso:

1. Require user valid-user: permite el acceso a cualquier usuario autenticado.
2. Require user alumne1 alumno2: permite el acceso a los usuarios indicados (alumno1 y alumno2).
3. Require group profes alumnos: permite el acceso a los usuarios que son miembros de alguno de los grupos indicados.

6 HTTPS

HTTPS

El protocolo HTTP sufre los mismos problemas de seguridad que los suyos compañeros de los inicios de internet (FTP, TFTP, SMTP...).

Toda la información viaja en texto limpio y es fácilmente monitorable por otros. Cuando un usuario se conecta a un web e indica el usuario y la contraseña, estos datos viajan sin ningún tipo de protección y cualquiera puede capturarlas. Si lo que se transmite son datos bancarios, listas de amistades íntimas o cualquier tipo de dato privado, es desaconsejable hacerlo por HTTP.

HTTPS

El primer mecanismo de seguridad que se implementó para HTTP fue el protocolo SSL (Secure Socket Layer o capa de zócalo seguro), desarrollado por Netscape. SSL proporciona una capa entre la capa de transporte TCP y la capa de aplicación HTTP en la que los datos viajan cifrados. El HTTPS solo es uno esquema URI que indica la utilización de HTML más algún mecanismo de transporte cifrado, como SSL o TLS.

HTTPS

Cuando se utiliza HTTP con un protocolo cifrado como SSL o TLS se llama HTTPS (secure HTTP). Use el puerto 443.

El protocolo SSL se envió al IETF (Internet Engineering Task Force o equipo de ingeniería de internet, el órgano rector de internet) para la estandarización y, tras varios cambios, surgió el protocolo TLS (Transport Layer Security, seguridad de capa de transporte). El TLS proporciona las mismas condiciones de confidencialidad y autenticación en las transmisiones HTTP que SSL.

HTTPS

Una de las ventajas del HTTPS es que permite la confidencialidad entre ambos extremos de la comunicación, aunque sólo sea uno de los extremos lo que se ha autenticado. Este modelo es muy práctico cuando, por ejemplo, un cliente anónimo compra en una web autenticada. Cuando se quiere pagar los billetes de avión, interesa que los datos de la tarjeta de crédito viajen cifrados y que el receptor sea la compañía aérea y no una web falsa.

El uso de los certificados no es exclusivo para autenticar el servidor. Si es necesario, los clientes pueden ser autenticados. Por ejemplo, una web puede requerir que los clientes dispongan del certificado que les otorga derecho a acceder (expedido, por ejemplo, por la misma entidad).

HTTPS

Los pasos necesarios para implementar comunicaciones seguras que permiten a un navegador cliente (o un cliente, sea quien sea) conectarse vía HTTPS a una sede web son:

- Certificados digitales: el servidor web debe disponer de una clave privada y de un certificado digital.
- Módulo `mod_ssl`: es necesario tener instalado el paquete de software que proporciona las prestaciones SSL en el servidor y que la configuración activa cargue los módulos pertinentes.
- Configuración de su web segura: finalmente, es necesario establecer las directivas SSL apropiadas a su web que se desea configurar para hacerla accesible vía SSL.

Certificados. Servidores de certificados

El objetivo de las siguientes explicaciones es implementar conexiones seguras HTTPS en la web `www.iocdaw.cat` utilizando SSL como mecanismo de transporte cifrado.

Supondremos que el servidor dispone ya de una clave privada y de un certificado, con independencia de cómo se hayan obtenido. En concreto, en el subdirectorio `certs` del directorio base del servicio web está:

- `server.crt`: el archivo correspondiente al certificado o clave pública del servidor. Este archivo asegura a los clientes que se conectan a su web que el servidor es quien realmente dice ser.
- `server.key`: es el archivo con la clave privada del servidor. Este archivo se ha codificado con una passphrase o frase de paso por lo que cada vez que se inicialice el servidor web deberá entrar esta frase.

Certificados. Servidores de certificados

Cabe recordar que los navegadores clientes validarán la confianza que les merece el certificado contrastando su emisor con la lista de entidades certificadoras que tienen cargada. Si el emisor del certificado no está, habrá que dar pasos para incorporar el certificado en el navegador.

Estos pasos pueden ser:

- Admitir el certificado como válido cuando el navegador presenta la excepción de seguridad.
- Obtener el certificado de la entidad CA (certification authority o autoridad de certificación) que la ha generado e incorporar la entidad a la lista de entidades en que el navegador confía.

Certificados. Servidores de certificados

Generar un certificado autofirmado

A modo de recordatorio rápido, se puede generar una clave privada y un certificado autofirmado haciendo:

```
1 # openssl req new x509 nodes out server.crt keyout server. key
```


Configuración de Apache para usar SSL

El servidor web podrá usar SSL si dispone de los módulos que proporcionan la capacidad. En caso de no tenerlos, es necesario buscar en los repositorios de software habitual un paquete que proporcione el módulo apropiado, instalarlo y examinarlo el contenido. Usualmente, tanto el paquete como el módulo que proporcionan las prestaciones de tráfico seguro SSL se llaman **mod_ssl**.

```
1  # Buscar el paquet mod_ssl i instal·lar lo.
2  ioc@ioc:/# apt-cache search mod_ssl
3  libapache2-mod-gnutls – Apache module for SSL and TLS encryption with GnuTLS
4  libapache2-mod-nss – NSS-based SSL module for Apache2
5  python-mod-pywebsocket – WebSocket extension for Apache HTTP Server
6
7  ioc@ioc:/# apt-get install libapache2-mod-gnutls
8
9  ioc@ioc:/# a2enmod ssl
10 ioc@ioc:/# service apache2 restart
```

Configuración de Apache para usar SSL

Como puede ver, el paquete contiene, entre otros, un archivo de configuración específico llamado `ssl.conf` y un único módulo llamado `mod_ssl`. Tanto uno como el otro están en el directorio `mods-available`. El archivo de configuración específico del módulo SSL contiene directivas que configuran el funcionamiento global del tráfico SSL. Como dice Apache, mejor no tocar nada. Se puede observar que el módulo está cargado en la carpeta `mods-enabled`:

```
1 ioc@ioc:/# ls -l /etc/apache2/mods-enabled/ssl.*
2 lrwxrwxrwx 1 root root 26 abr 15 10:38 /etc/apache2/mods-enabled/ssl.conf ->
   ../mods-available/ssl.conf
3 lrwxrwxrwx 1 root root 26 abr 15 10:38 /etc/apache2/mods-enabled/ssl.load ->
   ../mods-available/ssl.load
```

Configuración de Apache para usar SSL

Como puede ver, el paquete contiene, entre otros, un archivo de configuración específico llamado `ssl.conf` y un único módulo llamado `mod_ssl`. Tanto uno como el otro están en el directorio `mods-available`. El archivo de configuración específico del módulo SSL contiene directivas que configuran el funcionamiento global del tráfico SSL. Como dice Apache, mejor no tocar nada. Se puede observar que el módulo está cargado en la carpeta `mods-enabled`:

```
1 ioc@ioc:/# ls -l /etc/apache2/mods-enabled/ssl.*
2 lrwxrwxrwx 1 root root 26 abr 15 10:38 /etc/apache2/mods-enabled/ssl.conf ->
   ../mods-available/ssl.conf
3 lrwxrwxrwx 1 root root 26 abr 15 10:38 /etc/apache2/mods-enabled/ssl.load ->
   ../mods-available/ssl.load
```

Configuración de la web para usar SSL

Finalmente es necesario aplicar en su web `www.iocdaw.cat` las directivas SSL apropiadas para hacer posible el acceso a esta sede web por HTTPS. El listado de la directiva `VirtualHost` es:

```
1 <VirtualHost www.iocdaw.cat:443>
2   ServerAdmin admin@ www.iocdaw.cat
3   DocumentRoot /var/www/m08
4   ServerName www.iocdaw.cat
5   ErrorLog logs/m08 error_log
6   CustomLog logs/m08 access_log common SSLEngine On
7   SSLProtocol all  SHAv2
8   SSLCertificateKeyFile /var/www/certs/server.key
9   SSLCertificateFile /var/www/certs/server.crt
10  #SSLCACertificateFile /var/www/certs/ca.crt
11 </VirtualHost>
```

Configuración de la web para usar SSL

Las directivas usadas son:

- Puerto 443: éste es el puerto usual para las conexiones seguras HTTP. Si su web sólo escucha por este puerto sólo se podrá acceder a su contenido por HTTPS. Si se quieren sus diferentes para el tráfico cifrado y para el no cifrado basta con crear otra sede virtual con otro puerto.
- SSLEngine On: esta directiva indica que es necesario activar el tráfico SSL por en esta sede web.
- SSLProtocolall-SHAv2: en esta directiva se indican qué protocolos se pueden utilizarse para generar el tráfico cifrado. Las opciones all y -SHAv2 indican que se aceptan todos los protocolos válidos salvo el protocolo SHA versión 2.
- SSLCertificateKeyFile <clave privada del servidor>: esta directiva indica el archivo con la clave privada del servidor.

Configuración de la web para usar SSL

Las directivas usadas son:

- `SSLCertificateFile <certificado>`: indica cuál es el archivo que contiene el certificado del servidor. Éste es el certificado que los navegadores clientes verán y del que tendrán que decidir si confían o no.

- `SSLCACertificateFile <certificado-CA>`: esta directiva es opcional y permite indicar cuál es el archivo que contiene el certificado público que ha emitido la entidad de certificación CA.

Resumamos, si el certificado del servidor debe haber sido emitido por una entidad externa (por ejemplo, VeriSign), esta directiva permite que el cliente obtenga el certificado de la entidad emisora, ahorrándole la búsqueda. Ahora bien, todavía falta que el navegador cliente confíe en esta entidad.

Configuración de la web para usar SSL

Una vez configurada apropiadamente la sede virtual, es necesario poner de nuevo en funcionamiento el servicio web (momento en el que se pedirá la frase de paso del servidor por en la clave privada de `www.iocdaw.cat`). Desde cualquier navegador se debe poder acceder a la sede usando HTTPS. Ahora bien, se generará una excepción de seguridad para que el navegador desconoce la procedencia del certificado. Si el usuario acepta confiar en su web, el certificado se incorporará al navegador y accederá de forma cifrada en su sede web.

Otra opción es cargar previamente en el navegador el certificado de la entidad CA VerdadAbsoluta, que es quien ha actuado en el ejemplo como entidad certificadora local. Si esto se hace antes de contactar con su web, cuando el navegador acceda a ella por HTTPS ya no se producirá una excepción de seguridad. Debido a que el certificado del servidor está firmado por una entidad en la que el navegador confía (forma parte de su lista de trusted CAs) se aceptará automáticamente.

Verificación de las conexiones SSL

Los principales problemas que se pueden encontrar los navegadores al conectar con sus web con certificados son:

- Con un certificado autosignado no es necesario definir ninguna CA. El navegador cliente mostrará la típica pantalla de excepción de seguridad y habrá que indicar que se acepta el certificado de servidor para la entidad `www.iocdaw.cat`. Es un certificado emitido por la propia entidad.
- Con un certificado generado por una CA local es necesario incorporar manualmente el certificado en el navegador. Una vez hecho esto el navegador será capaz de validar el certificado del servidor con la CA que lo ha expedido (issuer). En el nuestro ejemplo, el certificado del servidor lo expide la CA VeritatAbsoluta.

Verificación de las conexiones SSL

Además de los navegadores, existen herramientas de entorno de texto para verificar conexiones SSL, de la misma forma que se utiliza telnet host 80 para verificar conexiones HTTP. Por un lado, se puede usar el propio OpenSSL y, por otro, se puede instalar la utilidad Curl, que permite realizar un amplio seguimiento del diálogo SSL.

```
1 OpenSSL> s_client -connect www.iocdaw.cat:443 -state -debug
2 curl https://www.iocdaw.cat -kv
```

7 LAMP

LAMP

LAMP, una pila de eficacia probada de **Linux, Apache, MySQL y PHP**.
Vamos a ver cómo podemos instalar esa pila junto con phpMyAdmin. Ya tenemos Apache instalado, ahora vamos a instalar mysql: **sudo apt install mysql-server**

```
daw@daw:~$ sudo apt install mysql-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Se instalarán los siguientes paquetes adicionales:
  libcgi-fast-perl libcgi-pm-perl libclone-perl libencode-locale-perl libevent-pthreads-2.1-7
  libfcgi-bin libfcgi-perl libfcgi0ldb1 libhtml-parser-perl libhtml-tagset-perl
  libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl
  liblwp-mediatypes-perl libmecab2 libprotobuf-lite23 libtimedate-perl liburi-perl mecab-ipadic
  mecab-ipadic-utf8 mecab-utils mysql-client-8.0 mysql-client-core-8.0 mysql-common
  mysql-server-8.0 mysql-server-core-8.0
Paquetes sugeridos:
  libdata-dump-perl libipc-sharedcache-perl libbusiness-isbn-perl libwww-perl mailx tinyca
Se instalarán los siguientes paquetes NUEVOS:
  libcgi-fast-perl libcgi-pm-perl libclone-perl libencode-locale-perl libevent-pthreads-2.1-7
  libfcgi-bin libfcgi-perl libfcgi0ldb1 libhtml-parser-perl libhtml-tagset-perl
  libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl
  liblwp-mediatypes-perl libmecab2 libprotobuf-lite23 libtimedate-perl liburi-perl mecab-ipadic
  mecab-ipadic-utf8 mecab-utils mysql-client-8.0 mysql-client-core-8.0 mysql-common mysql-server
  mysql-server-8.0 mysql-server-core-8.0
0 actualizados, 28 nuevos se instalarán, 0 para eliminar y 41 no actualizados.
Se necesita descargar 29,4 MB de archivos.
Se utilizarán 242 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
```

LAMP

Ahora vamos a indicar cuál es la contraseña para el usuario root de mysql. Para eso vamos a iniciar el cliente de mysql: **sudo mysql**
Y vamos a establecer la contraseña de root con el siguiente comando de SQL: **ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password by 'Patata23';**

```
daw@daw:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.31-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password by 'Patata23';
Query OK, 0 rows affected (0,01 sec)

mysql> _
```

sudo mysql_secure_installation

```
daw@daw:~$ cd /var/www/sitio1
daw@daw:/var/www/sitio1$ sudo mysql_secure_installation

Securing the MySQL server deployment.

Enter password for user root:

VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No: y

There are three levels of password validation policy:

LOW      Length >= 8
MEDIUM  Length >= 8, numeric, mixed case, and special characters
STRONG Length >= 8, numeric, mixed case, special characters and dictionary      file
```

LAMP

```
The 'validate_password' component is installed on the server.  
The subsequent steps will run with the existing configuration  
of the component.  
Using existing password for root.  
  
Estimated strength of the password: 50  
Change the password for root ? ((Press y|Y for Yes, any other key for No) : n  
  
... skipping.  
By default, a MySQL installation has an anonymous user,  
allowing anyone to log into MySQL without having to have  
a user account created for them. This is intended only for  
testing, and to make the installation go a bit smoother.  
You should remove them before moving into a production  
environment.  
  
Remove anonymous users? (Press y|Y for Yes, any other key for No) : y  
Success.  
  
Normally, root should only be allowed to connect from  
'localhost'. This ensures that someone cannot guess at  
the root password from the network.  
  
Disallow root login remotely? (Press y|Y for Yes, any other key for No) : y  
Success.
```

LAMP

```
By default, MySQL comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.
```

```
Remove test database and access to it? (Press y|Y for Yes, any other key for No) : y  
- Dropping test database...  
Success.
```

```
- Removing privileges on test database...  
Success.
```

```
Reloading the privilege tables will ensure that all changes made so far will take effect immediately.
```

```
Reload privilege tables now? (Press y|Y for Yes, any other key for No) : y  
Success.
```

```
All done!
```

LAMP

Instalar PHP

`sudo apt-get install php libapache2-mod-php php-mysql`

```
daw@daw:/var/www/sitio1$ sudo apt-get install php libapache2-mod-php php-mysql
[sudo] password for daw: _
```

Podemos comprobar que php está instalado con el comando:

`php -v`

```
daw@daw:/var/www/sitio1$ php -v
PHP 8.1.2-1ubuntu2.8 (cli) (built: Nov  2 2022 13:35:25) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.1.2, Copyright (c) Zend Technologies
    with Zend OPcache v8.1.2-1ubuntu2.8, Copyright (c), by Zend Technologies
daw@daw:/var/www/sitio1$
```



LAMP

Ahora podemos crear el archivo **prueba.php** en la raíz de nuestro sitio web con el siguiente contenido:

```
GNU nano 6.2 prueba.php *
<?php
    phpinfo();
?>
```

Luego accedemos a la URL:

<https://www.sitio1.com/prueba.php>

PHP Version 8.1.2-1ubuntu2.8

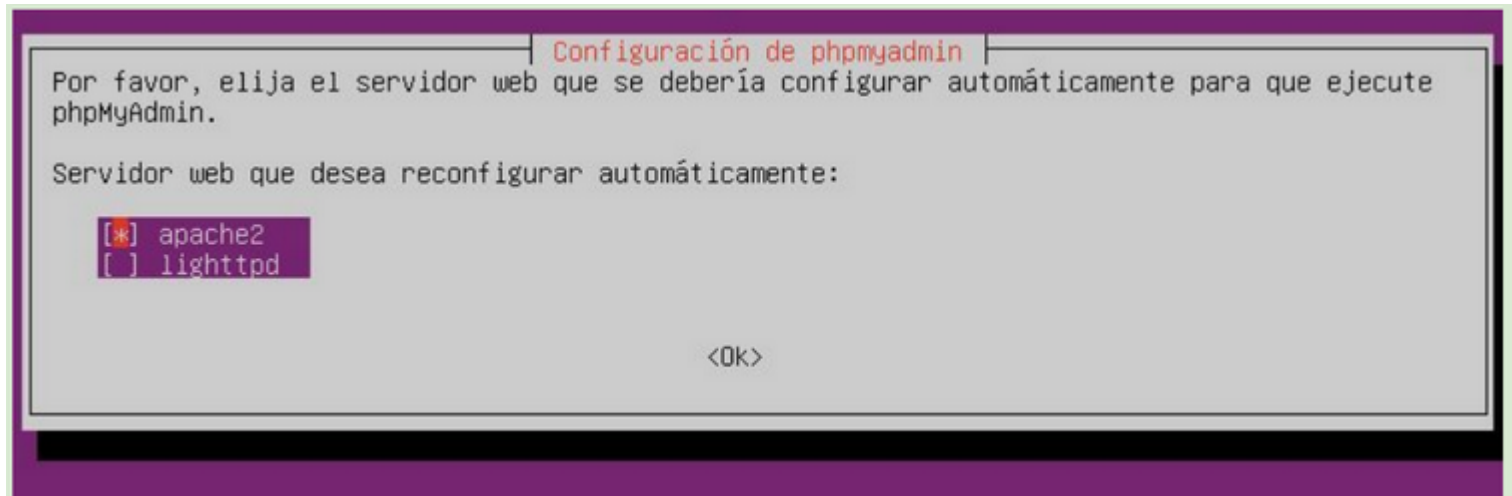
System	Linux daw 5.15.0-52-generic #58-Ubuntu SMP Thu Oct 13 08:03:55 UTC 2022 x86_64
Build Date	Nov 2 2022 13:35:25
Build System	Linux

LAMP

Ahora vamos a instalar phpMyAdmin, para ello ejecutaremos el siguiente comando:

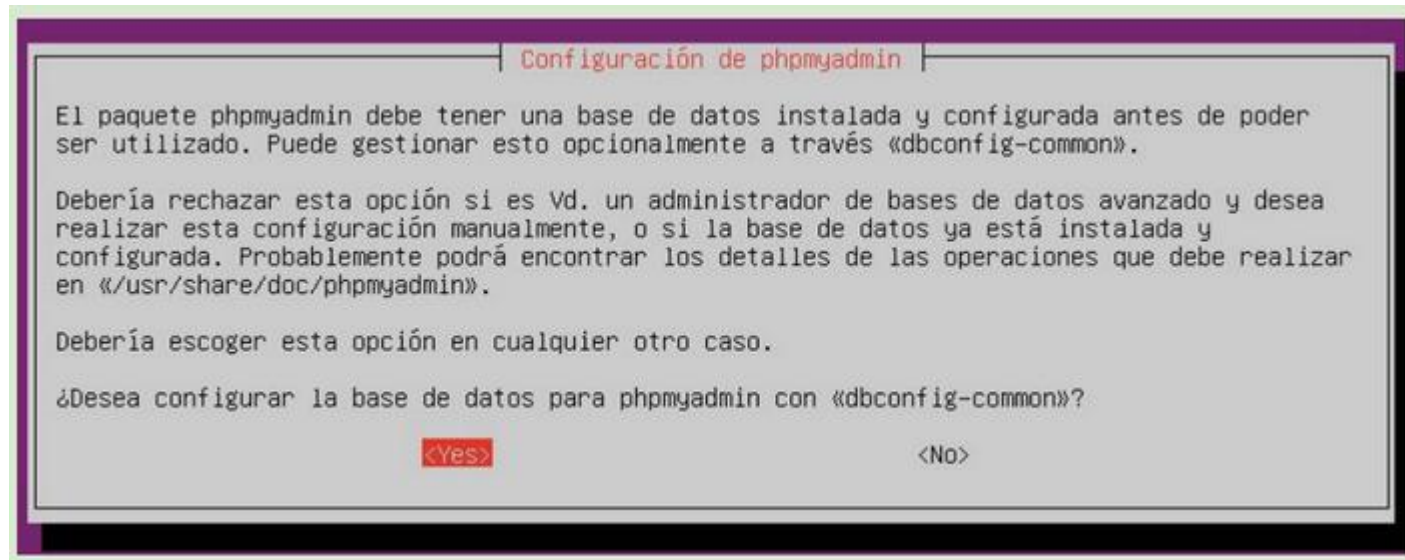
```
sudo apt-get install phpmyadmin php-mbstring php-zip php-gd php-json php-curl
```

Durante la instalación nos va a pedir en qué servidor web se va a instalar. Elegimos Apache.



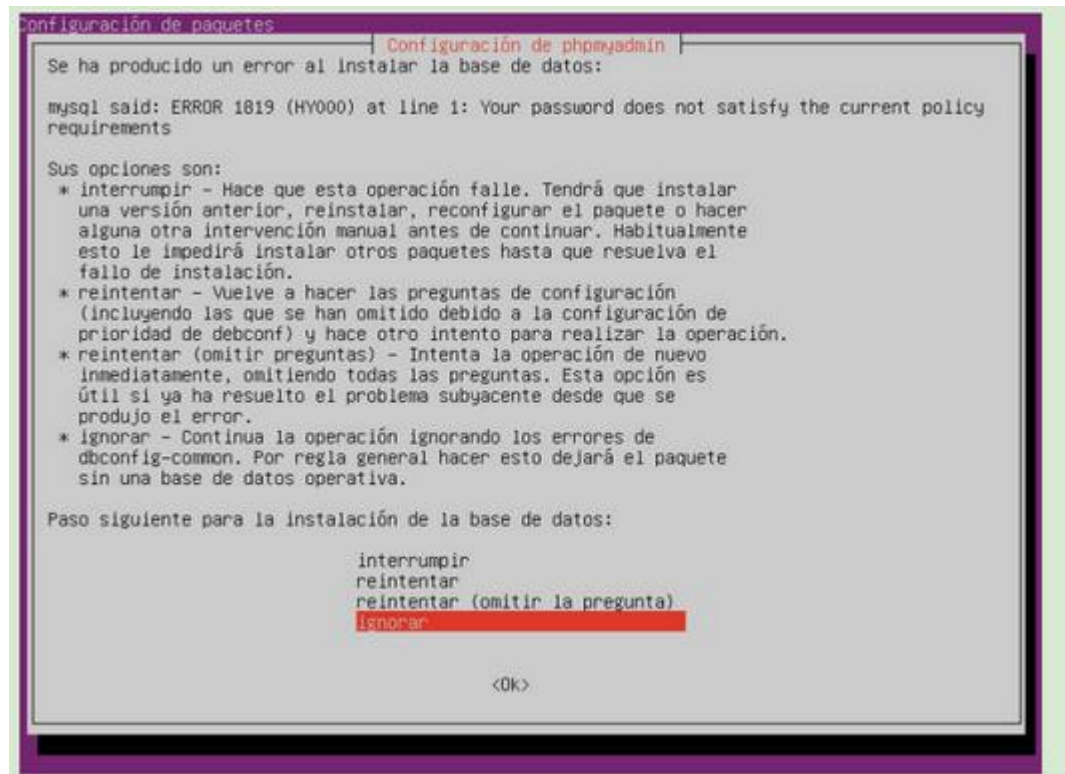
LAMP

A continuación nos va a pedir si queremos proporcionar una base de datos con la que trabajar. Indicamos que sí y proporcionamos el password del usuario root para la conexión.



LAMP

También nos puede dar un error si la contraseña no es lo suficientemente segura. Podemos darle a ignorar, aunque no es lo recomendado



```
Configuración de paquetes
Configuración de phpmyadmin
Se ha producido un error al instalar la base de datos:

mysql said: ERROR 1819 (HY000) at line 1: Your password does not satisfy the current policy
requirements

Sus opciones son:
* interrumpir - Hace que esta operación falle. Tendrá que instalar
una versión anterior, reinstalar, reconfigurar el paquete o hacer
alguna otra intervención manual antes de continuar. Habitualmente
esto le impedirá instalar otros paquetes hasta que resuelva el
fallo de instalación.
* reintentar - Vuelve a hacer las preguntas de configuración
(incluyendo las que se han omitido debido a la configuración de
prioridad de debconf) y hace otro intento para realizar la operación.
* reintentar (omitir preguntas) - Intenta la operación de nuevo
inmediatamente, omitiendo todas las preguntas. Esta opción es
útil si ya ha resuelto el problema subyacente desde que se
produjo el error.
* ignorar - Continúa la operación ignorando los errores de
dbconfig-common. Por regla general hacer esto dejará el paquete
sin una base de datos operativa.

Paso siguiente para la instalación de la base de datos:

interrumpir
reintentar
reintentar (omitir la pregunta)
ignorar


<Ok>
```

LAMP

phpMyAdmin

🛡️ 🔒 🔑 <https://www.sitio1.com/phpmyadmin/> ☆

Se ha modificado el archivo header.twig dentro de /usr/share/phpmyadmin/templates


Bienvenido a phpMyAdmin

Idioma - *Language*

Español - Spanish ▼

Iniciar sesión ⓘ

Usuario:

Contraseña:

Continuar

LAMP

phpMyAdmin

The screenshot displays the phpMyAdmin web interface in a browser. The address bar shows the URL: `https://www.sitio1.com/phpmyadmin/index.php?route=/&route=%2F`. The interface includes a top navigation bar with tabs for 'Bases de datos', 'SQL', 'Estado actual', 'Cuentas de usuarios', 'Exportar', 'Importar', 'Configuración', and 'Regi...'. A message banner states: 'Se ha modificado el archivo header.twig dentro de /usr/share/phpmyadmin/templates'. The left sidebar shows a tree view of databases: 'Nueva', 'information_schema', 'mysql', 'performance_schema', and 'sys'. The main content area is divided into two sections: 'Configuraciones generales' and 'Configuraciones de apariencia'. The 'Configuraciones generales' section includes a 'Cambio de contraseña' link, a 'Server connection collation' dropdown set to 'utf8mb4_unicode_ci', and a 'Más configuraciones' link. The 'Configuraciones de apariencia' section includes an 'Idioma - Language' dropdown set to 'Español - Spanish' and a 'Tema' dropdown set to 'pmahomme'. On the right, there are two summary boxes: 'Servidor de base de datos' and 'Servidor web'. The 'Servidor de base de datos' box lists: 'Servidor: Localhost via UNIX socket', 'Tipo de servidor: MySQL', 'Conexión del servidor: No se está utilizando', 'Versión del servidor: 8.0.31-0ubuntu0.22.0', 'Versión del protocolo: 10', 'Usuario: phpmyadmin@localhost', and 'Conjunto de caracteres del servidor: UTF-8'. The 'Servidor web' box lists: 'Apache/2.4.52 (Ubuntu)'.

Se ha modificado el archivo header.twig dentro de /usr/share/phpmyadmin/templates

Configuraciones generales

- Cambio de contraseña
- Server connection collation: utf8mb4_unicode_ci
- Más configuraciones

Configuraciones de apariencia

- Idioma - Language: Español - Spanish
- Tema: pmahomme

Servidor de base de datos

- Servidor: Localhost via UNIX socket
- Tipo de servidor: MySQL
- Conexión del servidor: No se está utilizando
- Versión del servidor: 8.0.31-0ubuntu0.22.0
- Versión del protocolo: 10
- Usuario: phpmyadmin@localhost
- Conjunto de caracteres del servidor: UTF-8

Servidor web

- Apache/2.4.52 (Ubuntu)