

INSTALACIÓN Y CONFIGURACIÓN

SERVIDOR WEB APACHE UBUNTU SERVER 20.04



INSTALACIÓN Y CONFIGURACIÓN SERVIDOR WEB APACHE

Apache es una fundación que proporciona soporte a la comunidad con un numeroso conjunto de proyectos *open source*. Uno de estos proyectos es el HTTP Server, servidor web *open source* para sistemas operativos basados en Unix y Windows NT.

En la actualidad es el servidor web más popular en Internet (seguido muy de cerca por Nginx).

Como siempre, comenzaremos actualizando las listas de paquetes desde los servidores y a continuación instalaremos, no sólo el paquete de apache2 sino su documentación:

```
sudo apt-get install apache2 apache2-doc
```



COMPROBACIÓN SERVIDOR WEB

Una vez terminado el proceso de instalación podemos comprobar que todo ha ido bien abriendo un navegador y solicitando la página de prueba que trae Apache.

- Tecleamos en la barra de direcciones del navegador (en el propio servidor web): <http://localhost>
- A continuación, desde otra máquina comprobaremos la conectividad de nuestra red interna, para ello teclearemos en la barra de direcciones del navegador la dirección IP del servidor Web.
- Por último, añadiremos el servidor web a los archivos de configuración de nuestro servidor de nombres de dominio, para finalmente comprobar, desde la barra de direcciones del navegador de un cliente, el correcto funcionamiento del servidor web.

<http://www.aulaSMR.com>

CONFIGURACIÓN DE APACHE

Los archivos de configuración de apache2 se encuentran en la carpeta ***/etc/apache2***.

El archivo principal de configuración es ***/etc/apache2/apache2.conf***. Este fichero es un simple archivo de texto, modificable con cualquier editor.

Es recomendable hacer una copia de seguridad del fichero antes de realizar modificaciones.

Como incluye muchas líneas con comentarios, tras inspeccionar su contenido, elimina los comentarios con *sed* (opcional).

Como siempre, cada vez que realicemos cambios en los que estemos interesados resulta imprescindible reiniciar el servidor para que estos surtan efecto.

/etc/init.d/apache2 restart

service apache2 restart

FICHEROS Y DIRECTORIOS DE CONFIGURACIÓN

Apache se configura colocando directivas en archivos de configuración de texto plano. Estas directivas están separadas entre los siguientes archivos y directorios:

apache2.conf: el archivo de configuración principal de Apache2. Contiene opciones que son globales para Apache2.

httpd.conf: históricamente el archivo principal de configuración de Apache2, llamado así por el servicio httpd. Actualmente el archivo no existe. En versiones anteriores puede estar presente, pero vacío, ya que todas las opciones de configuración se han movido a otros directorios.

envvars: archivo en el que se establecen las variables de entorno de Apache2.

mods-available: este directorio contiene los archivos de configuración para cargar módulos y configurarlos. No obstante, no todos los módulos tienen archivos de configuración específicos.

FICHEROS Y DIRECTORIOS DE CONFIGURACIÓN

mods-enabled: mantiene enlaces simbólicos a los archivos situados en */etc/apache2/mods-available*. Cuando se cree aquí un enlace simbólico al archivo de configuración de un módulo, éste se activará la próxima vez que se inicie apache2.

ports.conf: almacena las directivas que determinan los puertos TCP por los que Apache2 está escuchando.

sites-available: este directorio tiene archivos de configuración para los Servidores Virtuales. Los servidores virtuales permiten que Apache2 sea configurado para múltiples sitios que tengan configuraciones separadas.

sites-enabled: como **mods-enabled**, **sites-enabled** contiene enlaces simbólicos al directorio */etc/apache2/sites-available*. Similarmente cuando un archivo de configuración en **sites-available** tiene un enlace simbólico, el sitio configurado por él será activo una vez que se reinicie Apache.

CONFIGURACIÓN POR DEFECTO DE APACHE

La versión actual de Apache2 viene con una configuración por defecto adaptada a la creación de virtual host. En este caso, viene configurado con un virtual host único, usando la directiva Virtual Host, la cual puede ser modificada o usado como una plantilla para otros posibles virtual host que queramos tener.

Si lo dejamos tal cual, el virtual host por defecto será servido como nuestro sitio web.

Si queremos modificar el virtual host que trae por defecto Apache2 tendremos que modificar el fichero `/etc/apache2/sites-available/000-default.conf`.

```
<VirtualHost *:80>

    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

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

</VirtualHost>
```

CONFIGURACIÓN POR DEFECTO DE APACHE

El conjunto de directivas para un servidor virtual solo se aplican a un servidor virtual particular. Si una directiva se establece a nivel del servidor completo y no se define dentro de las opciones del servidor virtual, entonces se usarán las opciones predeterminadas.

Por ejemplo, puede definir una dirección de correo electrónico para el webmaster y no definir direcciones de correo individuales para cada servidor virtual.



DIRECTIVAS DE APACHE2

ServerAdmin: especifica la dirección de correo electrónico del administrador, esta dirección aparece en los mensajes de error, para permitir al usuario notificar un error al administrador.

La directiva **ServerName** es opcional y especifica el nombre FQDN por el cual va a responder tu sitio web.

El virtual host por defecto no lleva la directiva por tanto responde a todas las peticiones que no correspondan con la directiva **ServerName** especificada en otro [virtual Host](#).

Ejemplos de hosts virtuales: <https://httpd.apache.org/docs/2.4/vhosts/examples.html>

PINCHAME

DIRECTIVAS DE APACHE2

DocumentRoot: [directorio raíz](#) en el cual se va a buscar la página especificada como página de inicio.

- Por defecto el directorio de inicio es */var/www/html/* tal como indica el fichero */etc/apache2/sites-available/000-default.conf*. Por eso ahí encontraremos la página de prueba de Apache.
- Debemos tener en cuenta que al modificar la entrada **DocumentRoot** debemos modificar la directriz **<Directory /var/www/html/>**, la cual especifica características para este directorio.

Directivas contenedoras:

AllowOverride None: deshabilitamos el uso de ficheros .htaccess en el directorio especificado en la directiva Directory correspondiente.

Options Indexes: muestra una lista formateada de los contenidos de los directorios, si no existe DirectoryIndex (como index.html, index.php, etc.) en el directorio solicitado.

ALOJAMIENTO VIRTUAL

¿Que ocurre si deseas tener varios dominios separados, por ejemplo, miempresa.com y miempresa.es, atendidos con el mismo servidor?

Si todas las páginas están en la misma carpeta, ¿como sabe Apache que pagina servir para cada dominio?

Un host virtual, o VirtualHost, en Apache nos permite mantener múltiples nombres de host en nuestro servidor.



ALOJAMIENTO VIRTUAL

Apache divide su funcionalidad y componentes en unidades independientes que pueden ser configuradas de forma diferenciada. La unidad básica que describe un sitio individual o dominio se llama virtual host.

Estas asignaciones permiten al administrador utilizar un servidor para alojar varios dominios o sitios en una simple interfaz de red o IP utilizando un mecanismo de coincidencias. Esto es relevante para cualquiera que busque alojamiento para más de un sitio en un solo VPS (*Virtual Private Server*).

Cada dominio que es configurado apuntará al visitante a una carpeta específica que contiene la información del sitio, nunca indicará que el mismo servidor es responsable de otros sitios. Este esquema es expandible sin límites de software tanto como el servidor pueda soportar la carga.

PASO 1 - CREAR LA ESTRUCTURA DEL DIRECTORIO

El primer paso que necesitamos es crear la estructura de directorios que mantendrán la información de nuestro sitio.

Nuestro documento raíz (el directorio principal del host virtual en el cual Apache busca el contenido para mostrar) será configurado en directorios individuales dentro de la ruta `/var/www`. Crearemos los directorios aquí para los dos virtual hosts que pretendemos configurar en este ejemplo.

Dentro de cada uno de estos directorios crearemos un directorio denominado *public_html* el cual mantendrá la información pública del sitio y sus respectivos archivos. Esto nos dará más flexibilidad en nuestro alojamiento.

```
sudo mkdir -p /var/www/miempresa.com/public_html
```

```
sudo mkdir -p /var/www/miempresa.es/public_html
```

PASO 2 – ASIGNAR PERMISOS

Ahora tenemos la estructura de los directorios para nuestros archivos, pero el usuario **root** es el propietario de ellos. Si queremos que nuestro usuario pueda modificar los archivos en el directorio web, necesitamos cambiar el propietario haciendo lo siguiente:

```
sudo chown -R $USER:$USER /var/www/miempresa.com/public_html
```

```
sudo chown -R $USER:$USER /var/www/miempresa.es/public_html
```

La variable \$USER tomará el valor del usuario con el cual actualmente estás identificado. Al hacer esto, nuestro usuario ahora es propietario de los directorios public_html donde se almacenará nuestro contenido web.

PASO 2 – ASIGNAR PERMISOS

Debemos además modificar los permisos para asegurarnos que el permiso de lectura pueda ser aplicado a archivos y directorios para que las páginas puedan ser desplegadas correctamente:

```
sudo chmod -R 755 /var/www
```

El servidor ahora tiene los permisos necesarios para mostrar el contenido, y el usuario deberá ser capaz de crear contenido en los directorios a medida que sea necesario.

PASO 3 – CREAR UNA PÁGINA DE PRUEBA EN EL VH

Actualmente tenemos la estructura de directorios en su lugar. Vamos a añadir contenido web para mostrar, por ejemplo, podemos crear un archivo **index.html** para cada sitio con:

- Un título de la página (que se vea en la pestaña del navegador).
- Un texto categoría título 1 con un texto como – *Descripción de mi empresa*.
- Un párrafo con un texto descriptivo de la empresa.
- Copia el archivo en el directorio raíz del otro host virtual.
- A continuación, modifica el código **html** del archivo.

PASO 4 – CREAR NUEVOS ARCHIVOS VIRTUAL HOST

Los archivos Virtual Host son archivos que contienen información y configuración específica para el dominio y que le indican al servidor Apache como responden a las peticiones de varios dominios.

Apache incluye un archivo Virtual Host por defecto denominado 000-default.conf que podemos usar como plantilla. Realizaremos una copia para trabajar sobre ella y crear nuestro Virtual Host para cada dominio.

Iniciaremos con un dominio, configúralo, cópialo para el segundo dominio, y después realiza los ajustes necesarios. La configuración por defecto de Ubuntu requiere que cada archivo de configuración de Virtual Host termine en .conf.

CREACIÓN DEL ARCHIVO VIRTUAL HOST

Empezamos por copiar el archivo para el primer dominio:

```
sudo cp /etc/apache2/sites-available/000-default.conf /etc/apache2/sitesavailable/miempresa.com.conf
```

Abre el nuevo archivo con tu editor como super usuario:

```
sudo nano /etc/apache2/sites-available/miempresa.com.conf
```

Este archivo se verá como la imagen de la página 7 de este documento.

La configuración es bastante simple; personalizaremos los datos para nuestro primer dominio y agregaremos algunas directivas adicionales. Esta sección del Virtual Host coincide con cualquier petición que es solicitada al puerto 80, el puerto por defecto del protocolo HTTP.

CREACIÓN DEL ARCHIVO VIRTUAL HOST

Primero, necesitamos cambiar la directiva **ServerAdmin** por un correo del administrador del sitio que pueda recibir correos.

ServerAdmin admin@ejemplo.com

Después de esto, necesitamos agregar dos directivas. La primera llamada **ServerName**, que establece la base del dominio que debe coincidir para este Virtual Host. Esto será como tu dominio.

La segunda, llamada **ServerAlias**, determina nombres futuros que pueden coincidir y servirse como el nombre base o dominio principal. Esto es útil para host tipo www:

ServerName ejemplo.com

ServerAlias www.ejemplo.com

CREACIÓN DEL ARCHIVO VIRTUAL HOST

Lo que resta por cambiar para la configuración básica de un Virtual Host es la ubicación del directorio raíz para el dominio. Ya hemos creado la estructura de directorios que necesitamos, así que solo necesitamos modificar **DocumentRoot** para que apunte a dicho directorio:

DocumentRoot /var/www/miempresa.com/public_html

```
<VirtualHost *:80>

    ServerAdmin webmaster@miempresa.com
    DocumentRoot /var/www/miempresa.com/public_html

    ServerName miempresa.com
    ServerAlias www.miempresa.com

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

</VirtualHost>
```


CREACIÓN DEL SEGUNDO VIRTUAL HOST

Ahora que tenemos nuestro primer archivo Virtual Host configurado, podemos crear el segundo copiando el primero y realizando las modificaciones necesarias.

Empecemos por copiarlo:

```
sudo cp /etc/apache2/sites-available/miempresa.com.conf/ etc/apache2/sites-available/miempresa.es.conf
```

Abre el nuevo archivo con privilegios de super usuario en tu editor:

```
sudo vim /etc/apache2/sites-available/miempresa.es.conf
```

Ahora tenemos que modificar todas las directivas de información para referirnos al segundo dominio.

PASO 5 – HABILITA LOS NUEVOS HOSTS VIRTUALES

Ahora que hemos creado nuestros archivos virtual hosts, debemos habilitarlos. Apache incluye herramientas que nos permiten hacer esto.

Podemos usar la herramienta a2ensite para habilitar cada uno de nuestros sitios haciendo esto:

```
sudo a2ensite miempresa.com.conf
sudo a2ensite miempresa.es.conf
sudo a2dissite 000-default.conf #este último para deshabilitar el que trae por defecto, opcional
```

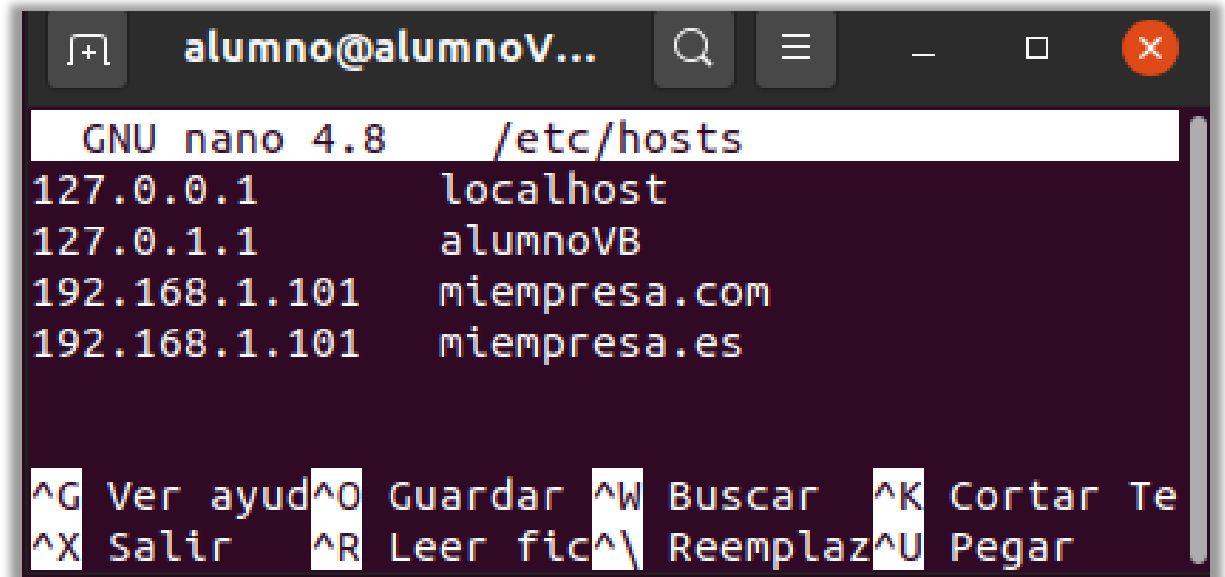
Cuando hayas concluido, deberás reiniciar Apache para asegurarte de que tus cambios surtan efecto.

PASO 6 – CONFIGURAR ARCHIVOS LOCALES

Si aún no estás utilizando nombres de dominio de tu propiedad para este procedimiento y utilizaste dominios ejemplo en su lugar, puedes al menos hacer pruebas de funcionalidad de este proceso modificando temporalmente el archivo hosts de la computadora desde donde realizas las comprobaciones.

Esto apuntará directamente cualquier petición para miempresa.com y miempresa.es en nuestra computadora y enviarlas a nuestro servidor en la IP correspondiente.

Esto es lo que queremos si no somos propietarios de esos dominios aún, solo con fines de prueba para nuestros Virtual Hosts.



The screenshot shows a terminal window titled 'alumno@alumnoV...' with a search bar and window controls. The terminal displays the GNU nano 4.8 editor editing the /etc/hosts file. The file content is as follows:

IP Address	Domain Name
127.0.0.1	localhost
127.0.1.1	alumnoVB
192.168.1.101	miempresa.com
192.168.1.101	miempresa.es

At the bottom of the terminal, a command palette is visible with the following shortcuts:

Shortcut	Action
^G	Ver ayuda
^O	Guardar
^W	Buscar
^K	Cortar Texto
^X	Salir
^R	Leer fichero
^Y	Reemplazar
^U	Pegar

SEGURIDAD BÁSICA EN APACHE

Para utilizar la seguridad básica de apache necesitamos activar el módulo de autenticación básica:

```
a2enmod auth_basic
```

Crear los usuarios y las contraseñas:

```
htpasswd -c /usr/local/apache2/conf/.htpasswd alumno
```

Esto nos creara el archivo **.htpasswd** en el directorio especifico y agregará el usuario **alumno** para el cual luego de presionar *enter* nos preguntara el password para este usuario.

Si necesitamos agregar otro usuario ejecutamos la misma instrucción pero sin el parámetro -c.

Las claves de los usuarios se crearan en forma encriptada por defecto dentro del archivo.

MECANISMOS DE CONTROL DE ACCESO

Apache proporciona distintos mecanismos para controlar el acceso de los usuarios a los recursos que presentamos mediante apache. En este caso vamos a utilizar la autenticación básica de Apache que limita todo tipo de acceso (GET, POST y HEAD) a un determinado directorio a través de un .htaccess (o nombre de fichero indicado), utilizando varias sentencias:

- La sentencia **AuthType Basic** indica el método de autenticación que deseamos usar. En nuestro caso el método de autenticación básico.
- **AuthName** indica el nombre que deseas darle al recurso. Este nombre aparecerá en la ventanita de autenticación donde el usuario tiene que introducir su nombre de usuario y contraseña, por lo que es recomendable darle un nombre descriptivo.
- **AuthUserFile** debe indicar la ubicación completa del fichero que contiene los datos de los usuarios (nombre de usuario y contraseña). Aunque habitualmente se le llame **.htpasswd**, puedes usar cualquier nombre.
- La sentencia **Require valid-user** indica que se limitará todo acceso ya sea de tipo GET, POST o HEAD y que solo se permitirá el acceso a los usuarios que aparezcan en el fichero **.htpasswd**.

MECANISMOS DE CONTROL DE ACCESO

Vamos a asegurar el servidor virtual de `www.miempresa.es`, para ello hay que poner las directivas **AuthType**, **AuthName** y **AuthUserFile** contenidas en la directiva **directory**, de la siguiente forma:

```
<VirtualHost *:80>

    ServerAdmin webmaster@miempresa.es
    DocumentRoot /var/www/miempresa.es/public_html

    ServerName miempresa.es
    ServerAlias www.miempresa.es

    <Directory /var/www/miempresa.es/public_html/>

        AuthType Basic
        AuthName "Acceso restringido, identifiquese"
        Require valid-user
        AuthUserFile /usr/local/apache2/conf/.htpasswd

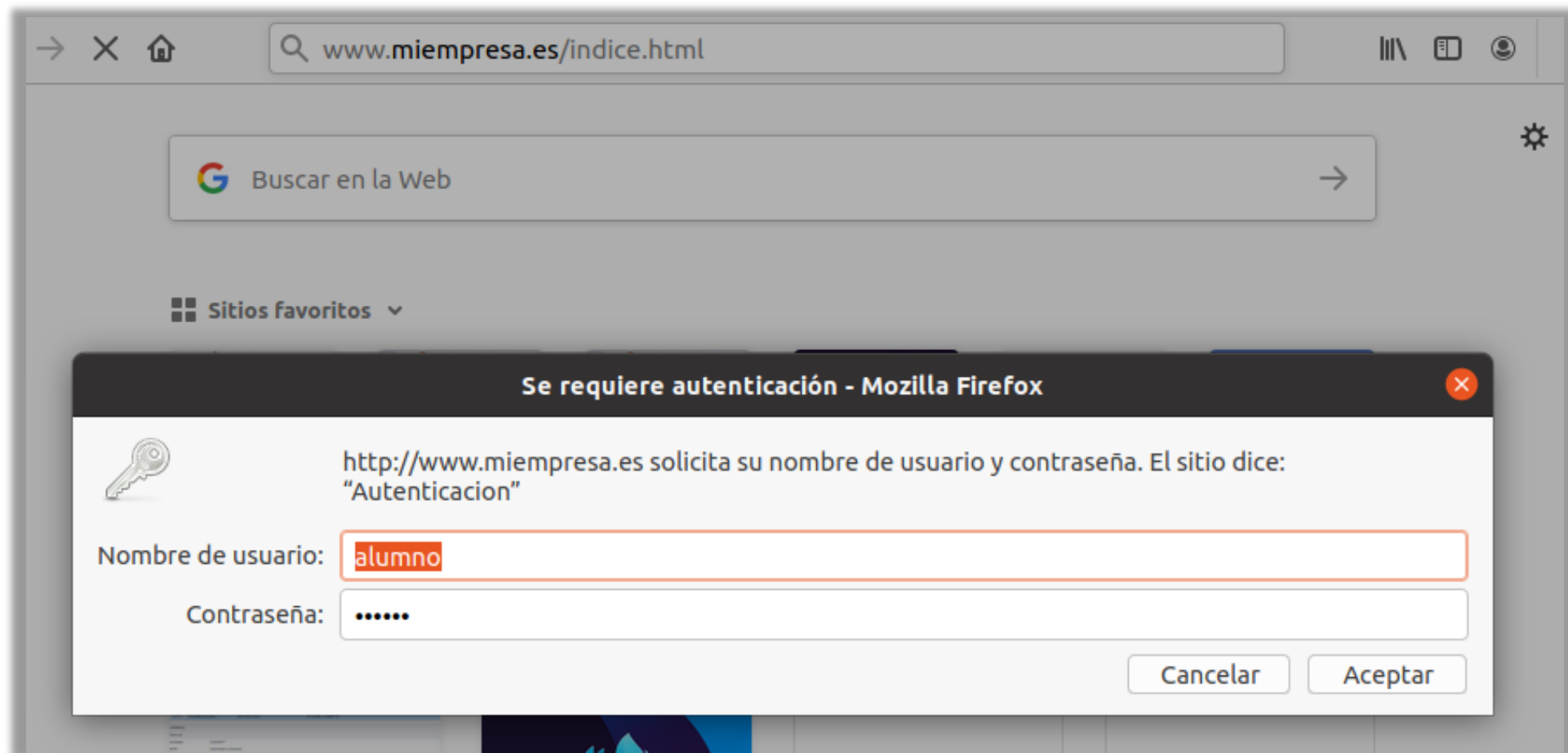
    </Directory>

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

</VirtualHost>
```

MECANISMOS DE CONTROL DE ACCESO

Al intentar acceder a la web que hemos protegido desde un navegador web tecleando en la barra de direcciones el ServerName (www.miempresa.es) el resultado es el siguiente:



OTRA POSIBILIDAD: USO DE HTACCESS

Lo primero es recordar que el uso de archivos **.htaccess** puede estar prohibido por una configuración anterior de Apache (directiva **AllowOverride None**).

Si esto ocurriese, no podríamos seguir con estos pasos. Para poder hacer todo lo que se plantea a continuación, es necesario dejar que Apache permita usar estos archivos, como mínimo para fijar aspectos de autenticación. Esto puede lograrse editando el fichero **000-default** (site por defecto, o el que corresponda) o **apache2.conf** y cambiar en la directiva <Directory> la entrada AllowOverride de None a AuthConfig.

Si hemos tenido que hacer esto, debemos reiniciar Apache posteriormente.

OTRA POSIBILIDAD: USO DE HTACCESS

A continuación, debemos ir al directorio web personal del usuario del sistema que queramos usar para esta prueba.

Creamos en ese directorio un fichero .htaccess con este contenido:

```
AuthType Basic
AuthName "Archivo restringido, identifíquese"
AuthUserFile /usr/local/apache2/conf/.htpasswd
Require valid-user
```

Y solo restaría reiniciar la máquina y probar la configuración como lo hemos hecho anteriormente.

INSTALACIÓN Y CONFIGURACIÓN SERVIDOR WEB APACHE

FIN