# LAMP Stack en Ubuntu Server

Implantación de Aplicaciones Web

José Juan Sánchez Hernández

Curso 2023/2024

# Índice

# Índice de figuras

# Índice de cuadros

# 1 LAMP Stack en Ubuntu Server

**LAMP** es el acrónimo usado para describir un sistema de infraestructura de Internet que usa las siguientes herramientas:

- Linux (Sistema Operativo)
- Apache (Servidor Web)
- MySQL/MariaDB (Sistema Gestor de Bases de Datos)
- PHP (Lenguaje de programación)

# 2 Linux

En esta práctica vamos a utilizar el sistema operativo Ubuntu Server.

## 2.1 Primeros pasos con: `apt`

`apt` (*Advanced Packaging Tool*) es el sistema gestor de paquetes utilizado en las distribuciones Debian y sus derivadas, como Ubuntu.

### 2.1.1 `apt update`

Actualiza la lista de paquetes:

```
sudo apt update
```

### 2.1.2 `apt upgrade`

Actualiza los paquetes instalados a sus últimas versiones disponibles. Los paquetes relacionados con el *kernel* no se actualizarán. Tampoco se resolverán los problemas de dependencias que necesiten eliminar otros paquetes.

```
sudo apt upgrade
```

### 2.1.3 `apt full-upgrade`

Realiza una actualización más completa del sistema operativo, como la actualización del *kernel*, resolución de dependencias entre paquetes, instalación y eliminación de paquetes si fuese necesario, etc.

También permite actualizar la versión del sistema operativo.

```
sudo apt full-upgrade
```

### 2.1.4 `apt install`

Instala un paquete determinado:

```
sudo apt install <package name>
```

### 2.1.5 apt remove

Desinstala un paquete:

```
sudo apt remove <package name>
```

### 2.1.6 apt purge

Desinstala un paquete determinado y elimina los archivos de configuración asociados:

```
sudo apt purge <package name>
```

### 2.1.7 apt search

Busca un paquete entre las descripciones de los paquetes:

```
sudo apt search <package name>
```

### 2.1.8 apt show

Muestra los detalles de un paquete:

```
sudo apt show <package name>
```

## 2.2 Instalación de un GUI Desktop

Ubuntu Server no cuenta con una interfaz gráfica de usuario (GUI) por defecto, pero si fuese necesario podemos instalar un GUI Desktop.

**Unity desktop:**

```
sudo apt install ubuntu-desktop
```

**GNOME desktop:**

```
sudo apt install ubuntu-gnome-desktop
```

**KDE desktop:**

```
sudo apt install kubuntu-desktop
```

Otros escritorios que podemos instalar son: `xubuntu-desktop`, `lubuntu-desktop`, `edubuntu-desktop`, etc.

## 2.3 Instalación de un servidor SSH

Ubuntu Server ya cuenta con un servidor SSH instalado por defecto, pero si fuese necesario instalarlo podemos hacerlo con el siguiente comando:

```
sudo apt install ssh
```

## 2.4 Cómo iniciar, parar y consultar el servicio de SSH

### 2.4.1 Método 1. `systemctl`

La mayoría de distribuciones Linux modernas utilizan `Systemd` como el sistema de inicio y administrador de servicios predeterminado.

```
sudo systemctl start ssh
sudo systemctl stop ssh
sudo systemctl status ssh
```

### 2.4.2 Método 2. `/etc/init.d/`

Alguna distribuciones Linux más antiguas pueden seguir utilizando `SysVinit`.

```
sudo /etc/init.d/ssh start
sudo /etc/init.d/ssh stop
sudo /etc/init.d/ssh status
```

## 2.5 Instalación de `git`

Ubuntu Server tiene instalado por defecto el sistema de control de versiones `git`, pero si fuese necesario instalarlo podemos hacerlo con el siguiente comando:

```
sudo apt install git
```

# 3 Apache

## 3.1 Instalación de Apache

```
sudo apt install apache2 -y
```

## 3.2 Cómo iniciar, parar y consultar el estado de Apache

### 3.2.1 Método 1. `systemctl`

```
sudo systemctl start apache2
sudo systemctl stop apache2
sudo systemctl restart apache2
sudo systemctl reload apache2
sudo systemctl status apache2
```

### 3.2.2 Método 2. `/etc/init.d/`

```
sudo /etc/init.d/apache2 start
sudo /etc/init.d/apache2 stop
sudo /etc/init.d/apache2 restart
sudo /etc/init.d/apache2 reload
sudo /etc/init.d/apache2 status
```

### 3.2.3 Método 3. `service`

```
sudo service apache2 start
sudo service apache2 stop
sudo service apache2 restart
sudo service apache2 reload
sudo service apache2 status
```

## 3.3 Archivos de configuración de Apache

Los archivos de configuración de Apache se almacenan en el directorio:

```
/etc/apache2/
```

En este directorio encontramos los siguientes archivos y directorios:

```
/* Archivos */├──
 apache2.conf├──
 envvars├──
 magic└──
 ports.conf

/* Directorios */├──
 conf-available├──
 conf-enabled├──
 mods-available├──
 mods-enabled├──
 sites-available└──
 sites-enabled
```

A continuación se describe brevemente cada uno de ellos:

- `apache2.conf`: Es el archivo de configuración principal. En este archivo se incluyen todos los archivos de configuración adicionales.
- `envvars`: Este archivo se definen las variables de entorno que hacen referencia al servidor web Apache y se utilizan en el archivo `apache2.conf`.
- `magic`: Este archivo contiene instrucciones para determinar el tipo de contenido o tipo MIME (MUltipurpose Internet Mail Extensions) de un archivo en función de los primeros bytes de un archivo. Los navegadores a menudo usan el tipo MIME (y no la extensión de archivo) para determinar cómo procesará un documento; por lo tanto, es importante que los servidores estén configurados correctamente para adjuntar el tipo MIME correcto al encabezado del objeto de respuesta. Puede encontrar más información sobre los tipo MIME aquí.
- `ports.conf`: En este archivo se definen los puertos TCP donde el servidor Apache estará escuchando peticiones.
- `conf-available`: Este directorio contiene archivos de configuración que se aplican a todos los hosts virtuales de forma global.
- `conf-enabled`: Este directorio contiene enlaces simbólicos a los archivos de configuración del directorio `conf-available` que están activos.
- `mods-available`: Este directorio contiene los archivos de configuración de los módulos que se pueden utilizar para añadir nuevas funcionalidades al servidor.
- `mods-enabled`: Este directorio contiene enlaces simbólicos a los archivos de configuración del directorio `mod-available` que están activos.
- `sites-available`: Este directorio contiene los archivos de configuración de los hosts virtuales.
- `sites-enabled`: Este directorio contiene enlaces simbólicos a los archivos de configuración del directorio `sites-available` que están activos.

## 3.4 Cómo modificar el puerto por defecto de Apache

Para configurar los puertos donde Apache escuchará las peticiones HTTP tenemos que modificar el archivo `/etc/apache2/ports.conf`.

Este es el contenido que tiene por defecto el archivo `/etc/apache2/ports.conf`.

```
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80

<IfModule ssl_module>
        Listen 443
</IfModule>

<IfModule mod_gnutls.c>
        Listen 443
</IfModule>
```

**Ejemplo:**

Vamos a modificar el puerto por defecto de Apache para que escuche las peticiones en el puerto `8000`.

Modificamos el archivo `/etc/apache2/ports.conf` y cambiamos el valor de la directiva `Listen`.

```
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 8000

<IfModule ssl_module>
        Listen 443
</IfModule>

<IfModule mod_gnutls.c>
        Listen 443
</IfModule>
```

Una vez modificado el archivo `/etc/apache2/ports.conf` tenemos que modificar el puerto en el archivo de configuración del sitio virtual por defecto: `/etc/apache2/sites-available/000-default.conf`.

En la directiva `<VirtualHost *:8000>` vamos a indicar que todas las peticiones que lleguen al puerto `8000` se les aplique la configuración que se encuentra dentro de este bloque.

```
<VirtualHost *:8000>
        #ServerName www.example.com
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html

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

```
8 </VirtualHost>
```

Una vez hecho los cambios en los archivos de configuración, tenemos que reiniciar el servicio de Apache para que los cambios se apliquen.

```
1 sudo systemctl restart apache2
```

**Nota:** No olvide comprobar que **el nuevo puerto que ha configurado está abierto en las reglas del firewall** y no se está bloqueando.

## 3.5 Cómo modificar el directorio por defecto de Apache

El directorio que utiliza Apache para servir contenido se configura en la directiva `DocumentRoot` y está configurada con el valor `/var/www/html` por defecto.

Esta directiva se puede configurar en los archivos de configuración de los sitios virtuales dentro del directorio `/etc/apache2/sites-available`.

A continuación se muestra un ejemplo con el contenido del archivo `/etc/apache2/sites-available/000-default.conf`.

```
1 <VirtualHost *:80>
2         #ServerName www.example.com
3         ServerAdmin webmaster@localhost
4         DocumentRoot /var/www/html
5
6         ErrorLog ${APACHE_LOG_DIR}/error.log
7         CustomLog ${APACHE_LOG_DIR}/access.log combined
8 </VirtualHost>
```

**Ejemplo:**

En este ejemplo vamos a modificar la directiva `DocumentRoot` para que el directorio que utilice Apache sea `/home/ubuntu/misitioweb`.

Creamos el nuevo directorio donde vamos a alojar la web.

```
1 mkdir -p /home/ubuntu/misitioweb
```

Creamos un archivo `index.html` de prueba.

```
1 echo "Hola mundo!" > /home/ubuntu/misitioweb/index.html
```

Modificamos el propietario y el grupo del directorio `/home/ubuntu/misitioweb` de forma recursiva, para que el usuario y grupo `www-data` puedan acceder a él. Este es el usuario con el que se ejecuta el servicio Apache.

```
1 sudo chown -R www-data:www-data /home/ubuntu/misitioweb
```

Modificamos los permisos del directorio donde está la web y el directorio que la contiene. Estos directorios tienen que tener habilitados los permisos de ejecución.

```
sudo chmod 755 /home/ubuntu
sudo chmod 755 /home/ubuntu/misitioweb
```

Configuramos el archivo donde está definido el sitio virtual por defecto: `/etc/apache2/sites-available/000-default.conf`.

```
<VirtualHost *:80>
        #ServerName www.example.com
        ServerAdmin webmaster@localhost
        DocumentRoot /home/ubuntu/misitioweb

        <Directory /home/ubuntu/misitioweb>
          Options Indexes FollowSymLinks
          AllowOverride None
          Require all granted
        </Directory>

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Hemos añadido un bloque `<Directory>` para configurar los permisos del directorio `/home/ubuntu/misitioweb`. En este bloque hemos configurado las siguientes opciones:

- `Options Indexes FollowSymLinks`: Permite listar directorios (`Indexes`) y seguir enlaces simbólicos (`FollowSymLinks`).
- `AllowOverride None`: No permite el uso de archivos `.htaccess` para que no se puedan aplicar directivas configuración en el directorio.
- `Require all granted`: Permite el acceso a todos los usuarios.

Una vez configurado el archivo es necesario reiniciar el servicio para que se apliquen los cambios:

```
sudo systemctl restart apache2
```

## 3.6 Cómo habilitar/deshabilitar un módulo de Apache

Para habilitar un módulo de Apache utilizamos el comando `a2enmod`.

**Ejemplo:**

En este ejemplo vamos a habilitar el módulo `rewrite` de Apache. Este módulo permite reescribir las URLs y configurar reglas de redireccionamiento, para hacer las URLS más amigables y mejorar el SEO de los sitios web.

```
sudo a2enmod rewrite
```

Una vez que hemos habilitado el módulo es necesario reiniciar el servicio de Apache para que se apliquen los cambios.

```
sudo systemctl restart apache2
```

Para deshabilitar un módulo utilizamos el comando `a2dismod`.

**Ejemplo:**

Para deshablitar el módulo `rewrite` ejecutamos el siguiente comando.

```
sudo a2dismod rewrite
```

Una vez que hemos habilitado el módulo es necesario reiniciar el servicio de Apache para que se apliquen los cambios.

```
sudo systemctl restart apache2
```

## 3.7 Cómo configurar la directiva `DirectoryIndex`

La directiva `DirectoryIndex` se utiliza para configurar el orden prioridad de los archivos que se van a mostrar cuando se accede a un directorio. Por ejemplo, si un directorio tiene los archivos `index.html` y `index.php`, el servidor enviará al usuario el archivo que aparezca antes en la lista de prioridad de la directiva `DirectoryIndex`.

Si la configuración fuese esta:

```
DirectoryIndex index.php index.html
```

El servidor enviaría al usuario el archivo `index.php`.

Esta directiva se puede configurar a nivel global en el archivo de configuración:

- `/etc/apache2/mods-available/dir.conf`

O también se puede configurar dentro de cada uno de los sitios virtuales en:

- `/etc/apache2/sites-available/`

**Ejemplo en el archivo `/etc/apache2/mods-available/dir.conf`:**

```
DirectoryIndex index.php index.html index.cgi index.pl index.php index.xhtml
    index.htm
```

**Ejemplo en un host virtual:**

```
<VirtualHost *:80>
    #ServerName www.example.com
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/

    DirectoryIndex index.php index.html

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

## 3.8 Cómo crear un nuevo host virtual

Los hosts virtuales se crean en el directorio `/etc/apache2/sites-available/`.

Para habilitar y deshabilitar un host virtual se utilizan los comandos `a2ensite` y `a2dissite`.

```
sudo a2ensite <nombre_del_archivo_de_configuración>
```

```
sudo a2dissite <nombre_del_archivo_de_configuración>
```

Donde `<nombre_del_archivo_de_configuración>` es el nombre del archivo de configuración del host virtual que se almacena en el directorio `/etc/apache2/sites-available/`.

### 3.8.1 Host virtual basado en el nombre de dominio

**Paso 1**

Vamos a crear un directorio para alojar cada uno de los sitios web.

```
sudo mkdir -p /var/www/html/web1
sudo mkdir -p /var/www/html/web2
```

Modificamos el propietario y el grupo de los directorios de forma recursiva.

```
sudo chown -R www-data:www-data /var/www/html
```

**Paso 2**

Creamos un archivo de prueba para cada uno de los sitios web.

```
echo "Sitio web 1" | sudo tee /var/www/html/web1/index.html
echo "Sitio web 2" | sudo tee /var/www/html/web2/index.html
```

**Paso 3**

Creamos los archivos de configuración para cada uno de los sitios web. Empezamos por el sitio web 1.

```
sudo nano /etc/apache2/sites-available/web1.conf
```

Creamos el contenido para el archivo de configuración del sitio web 1.

```
<VirtualHost *:80>
    ServerAdmin webmaster@web1.com
    ServerName web1.com
    DocumentRoot /var/www/html/web1
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Creamos los archivos de configuración para el sitio web 2.

```
sudo nano /etc/apache2/sites-available/web2.conf
```

Creamos el contenido para el archivo de configuración del sitio web 2.

```
<VirtualHost *:80>
    ServerAdmin webmaster@web2.com
    ServerName web2.com
    DocumentRoot /var/www/html/web2
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

**Paso 4**

Antes de habilitar los nuevos sitios virtuales vamos a deshabilitar el sitio virtual que viene por defecto.

```
sudo a2dissite 000-default.conf
```

Una vez deshabilitado el sitio virtual por defecto, habilitamos los dos sitios virtuales que hemos creado con el comando `a2ensite`.

```
sudo a2ensite web1.conf
sudo a2ensite web2.conf
```

**Paso 5**

Para que se aplique la nueva configuración es necesario ejecutar el comando:

```
sudo systemctl reload apache2
```

**Paso 6**

Para poder comprobar que los sitios virtuales están funcionando correctamente, vas a necesitar hacer alguna de estas dos opciones:

- (1) Registrar los dominios que hemos utilizado en el ejemplo en un proveedor de nombres de dominio para que apunten a la IP pública del servidor web.
- (2) Editar el archivo `/etc/hosts` de tu equipo para resolver los nombres de dominio de forma local en tu entorno de desarrollo.

En este ejemplo vamos a elegir la segunda opción, por lo tanto, vamos a editar el archivo `/etc/hosts` de nuestro equipo y vamos a añadir las siguientes líneas, sustituyendo `IP_SERVIDOR_WEB` por la IP pública del servidor web.

```
IP_SERVIDOR_WEB   web1.com
IP_SERVIDOR_WEB   web2.com
```

Por ejemplo, si la IP pública de nuestro servidor es `172.16.12.178` el archivo `/etc/hosts` quedaría de la siguiente forma:

```
172.16.12.178   web1.com
172.16.12.178   web2.com
```

### 3.8.2 Host virtual basado en el puerto

También es posible crear sitios virtuales que escuchen en puertos diferentes y que sirvan contenido diferente en función del puerto donde han recibido la petición.

**Ejemplo:**

En este ejemplo vamos a crear dos sitios virtuales que escuchen en los puertos `8000` y `80`. Cuando lleguen peticiones al puerto `8000` se servirá el sitio que está almacenado en el directorio `/var/www/html/web-port-8000` y cuando lleguen peticiones al puerto `80` se servirá el sitio que está almacenado en el directorio `/var/www/html/web-port-80`.

**Importante:** Observe que hemos añadido la directiva `Listen 8000` en el archivo de configuración del sitio virtual, no ha sido necesario modificar el archivo `/etc/apache2/ports.conf`.

```
Listen 8000
<VirtualHost *:8000>
    DocumentRoot /var/www/html/web-port-8000
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>

Listen 80
<VirtualHost *:80>
    DocumentRoot /var/www/html/web-port-80
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

## 3.9 Cómo ocultar la versión de Apache al cliente

Por razones de seguridad y privacidad, se considera una buena práctica ocultar al cliente la versión del servidor web que se está utilizando.

Las directivas que se utilizan para ocultar la versión de Apache son las siguientes:

- `ServerSignature`: Se utiliza para configurar si se debe incluir la versión del servidor en las páginas de error y en las páginas de índice de directorios.
- `ServerTokens`: Se utiliza para configurar el nivel de detalle de la información sobre el servidor que se incluye en las respuestas HTTP que se envían al cliente.

**Importante:**

- La directiva `ServerTokens` se tiene que aplicar a todo el servidor, no se puede aplicar a un host virtual específico. Sin embargo, la directiva `ServerSignature` sí se puede aplicar de formma independiente a un host virtual.

**Ejemplo:**

Una posible configuración de estas directivas sería la siguiente:

```
ServerSignature Off
ServerTokens Prod
```

**Ejemplo:**

Vamos a añadir las directivas `ServerSignature` y `ServerTokens` en el archivo donde está definido el sitio virtual por defecto: `/etc/apache2/sites-available/000-default.conf`.

```
ServerSignature Off
ServerTokens Prod

<VirtualHost *:80>
        #ServerName www.example.com
        ServerAdmin webmaster@localhost
        DocumentRoot /home/ubuntu/misitioweb

        <Directory /home/ubuntu/misitioweb>
          Options Indexes FollowSymLinks
          AllowOverride None
          Require all granted
        </Directory>

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Puede comprobar las cabeceras HTTP que se envían al cliente con el comando `curl`:

```
curl -Iv http://IP_SERVIDOR_WEB
```

- El parámetro `-I` indica que sólo queremos ver las cabeceras HTTP.
- El parámetro `-v` indica que queremos ver la salida detallada.
- Tendrá que sustituir el valor de `IP_SERVIDOR_WEB` por la dirección IP o el nombre de dominio del servidor web.

Puede encontrar más información sobre estas directivas en la documentación oficial:

- ServerSignature
- ServerTokens

## 3.10 Archivos de log de Apache

### 3.10.1 `access.log`

El servidor almacena en el registro de acceso información sobre todas las peticiones que procesa.

```
/var/log/apache2/access.log
```

### 3.10.2 `error.log`

El registro de errores del servidor.

```
/var/log/apache2/error.log
```

### 3.10.3 Mensajes de error más detallados en Apache

Para poder localizar un error debemos configurar que los archivos de log sean más detallados. Para ello, debemos cambiar o añadir la línea `LogLevel`en el archivo `/etc/apache2/apache2.conf`. Los argumentos posibles se detallan a continuación. Por ejemplo:

```
LogLevel debug
```

Existen varios niveles jerárquicos en los registros de error disponibles, cada uno de ellos identificado por su propia clave. El valor por defecto de `LogLevel` es `warn`. A continuación se describen los valores posibles, ordenados de manera descendente según su grado de importancia.

- `emerg`: Emergencias, no se puede utilizar el servidor.
- `alert`: Require acción inmediata.
- `crit`: Condiciones críticas.
- `error`: Condiciones de error.
- `warn`: Condiciones de advertencia.
- `notice`: Condicón normal pero significativa.
- `info`: Informativa.
- `debug`: Mensajes de corrección de errores.

`emerg` es el valor que graba el menor número de información y `debug` el que más. El problema es que `debug` graba mucha información que no tiene nada que ver con el problema, por lo que es conveniente cambiar el valor una vez que hayamos resuelto dicho problema.

# 4 MySQL Server

## 4.1 Instalación de MySQL Server

```
sudo apt install mysql-server -y
```

## 4.2 Cómo iniciar, parar y consultar el estado de MySQL Server

### 4.2.1 Método 1. `systemctl`

```
sudo systemctl start mysql
sudo systemctl stop mysql
sudo systemctl restart mysql
sudo systemctl status mysql
```

### 4.2.2 Método 2. `/etc/init.d/`

```
sudo /etc/init.d/mysql start
sudo /etc/init.d/mysql stop
sudo /etc/init.d/mysql restart
sudo /etc/init.d/mysql reload
sudo /etc/init.d/mysql force-reload
sudo /etc/init.d/mysql status
```

## 4.3 Archivos de configuración de MySQL Server

```
/etc/mysql/mysql.cnf
```

También podemos encontrar archivos de configuración en los directorios:

```
/etc/mysql/conf.d/
/etc/mysql/mysql.conf.d/
```

## 4.4 Archivos de log de MySQL Server

```
/var/log/mysql/error.log
```

## 4.5 Cómo acceder a MySQL Server desde consola con el usuario `root`

**Opción 1**

Una vez que hemos instalado MySQL Server en nuestro sistema vamos a acceder a la consola de MySQL. En primer lugar vamos a iniciar una sesión como `root`:

```
sudo su
```

Una vez que hemos iniciado una sesión como `root` vamos a iniciar la consola de MySQL también como `root` sin necesidad de indicarle ningún password (no es necesario usar el modificador `-p`).

```
mysql -u root
```

Una vez hecho esto ya tendríamos acceso a la consola de MySQL como `root`.

**Opción 2**

También es posible acceder a la consola de MySQL como `root` con el siguiente comando:

```
sudo mysql
```

## 4.6 Cómo cambiar la contraseña del usuario `root` (Método 1)

En primer lugar accedemos a la consola de MySQL como `root` y seleccionamos la base de datos **mysql**.

```
USE mysql;
```

Vamos a revisar los usuarios que existen en MySQL y qué método tienen establecido para autenticar.

```
SELECT User, Host, plugin FROM user;

+------------------+-----------+-----------------------+
| User             | Host      | plugin                |
+------------------+-----------+-----------------------+
| root             | localhost | auth_socket           |
| mysql.session    | localhost | caching_sha2_password |
| mysql.sys        | localhost | caching_sha2_password |
| debian-sys-maint | localhost | caching_sha2_password |
+------------------+-----------+-----------------------+
4 rows in set (0.00 sec)
```

Podemos ver que para el usuario `root@localhost` el método de autenticación es `auth_socket`. Esto quiere decir que el usuario `root` usará las mismas credenciales que tiene en el sistema operativo.

Si queremos cambiar la contraseña de `root` tendremos que cambiar el método de autenticación a `mysql_native_password` o `caching_sha2_password`. Tendrá que sustituir *`nueva_contraseña`* por la contraseña que desee.

**Opción `mysql_native_password` (Antigua)**

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '
    nueva_contraseña';
```

**Opción `caching_sha2_password` (Nueva)**

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH caching_sha2_password BY '
    nueva_contraseña';
```

La documentación oficial de MySQL indica que la nueva opción `caching_sha2_password` es más segura que la antigua opción `mysql_native_password` y que por lo tanto es la opción recomendada.

Después habrá que ejecutar el siguiente comando para que las actualizaciones realizadas tengan efecto.

```
FLUSH PRIVILEGES;
```

## 4.7 Cómo cambiar la contraseña del usuario `root` (Método 2 `--skip-grant-tables`)

Otra opción para modificar la contraseña del usuario `root` en MySQL es iniciar el servicio con la opción `--skip-grant-tables`, que permite que cualquier usuario se pueda conectar sin necesidad de realizar el proceso de autenticación.

En primer lugar detenemos el servicio de MySQL.

```
sudo systemctl stop mysql
```

Como vamos a iniciar el proceso de MySQL de forma manual vamos a necesitar crear de forma manual el directorio `/var/run/mysqld` y asignarle que el propietario de este directorio es `mysql` del grupo `mysql`.

```
sudo mkdir -p /var/run/mysqld
sudo chown mysql:mysql /var/run/mysqld
```

Una vez que hemos detenido el servicio de MySQL, lo volvemos a iniciar pero haciendo uso del comando que está en la ruta `/usr/sbin/mysqld` y pasándole como parámetro la opción `--skip-grant-tables` que permite conectarnos sin contraseña y con todos los privilegios. Esta opción deshabilita la gestión de usuarios y por lo tanto, no es posible utilizar sentencias como `ALTER USER` y `SET PASSWORD`. Al utilizar la opción `--skip-grant-tables` se habilita automáticamente la opción `--skip-networking` para desactivar las conexiones remotas como mecanismo de seguridad. Observe que hemos añadido & al final del comando para que el proceso se ejecute en segundo plano.

```
sudo /usr/sbin/mysqld --skip-grant-tables &
```

Podemos comprobar que le proceso se está ejecutando de forma correcta haciendo un listado de todos los procesos que están en ejecución con `ps aux` y buscando en la lista el nombre del proceso con `grep mysqld`.

```
ps aux | grep mysqld
```

Una vez que hemos comprobado que el proceso está en ejecución, podemos conectarnos al servidor de MySQL. No es necesario indicar ningún usuario ni contraseña, nos conectaremos automáticamente con todos los privilegios como `root`.

```
mysql
```

Ahora tendremos que indicarle al servidor que tiene que recargar las tablas encargadas de la autenticación de los usuarios para poder activar la gestión de usuarios. Recuerda que la opción `--skip-grant-tables` deshabilita esta funcionalidad.

```
FLUSH PRIVILEGES;
```

Modificamos la contraseña del usuario 'root'@'localhost', reemplazando `nueva_contraseña` por la contraseña que queramos asignar.

```
ALTER USER 'root'@'localhost' IDENTIFIED BY 'nueva_contraseña';
```

Salimos del cliente de MySQL.

```
exit;
```

Detenemos el proceso `mysqld`.

```
sudo pkill mysqld
```

Reiniciamos el servicio de MySQL.

```
sudo systemctl start mysql
```

## 4.8 Algunos comandos útiles para MySQL Server desde consola

### 4.8.1 Listar todas las bases de datos disponibles

```
SHOW DATABASES;
```

### 4.8.2 Crear una nueva base de datos

```
CREATE DATABASE <database>;
```

### 4.8.3 Borrar una base de datos

```
DROP DATABASE <database>;
```

### 4.8.4 Seleccionar una base de datos

```
USE <database>;
```

### 4.8.5 Listar las tablas que contiene una base de datos

```
SHOW TABLES;
```

### 4.8.6 Mostrar la estructura de una tabla

```
DESCRIBE <table>;
```

### 4.8.7 Cerrar la sesión y salir

```
exit
```

```
quit
```

## 4.9 Usuarios y permisos en MySQL Server desde consola

### 4.9.1 Eliminar un usuario

La sintaxis para eliminar un usuario en MySQL es la siguiente:

```
DROP USER [IF EXISTS] user [, user] ...
```

**Ejemplo:**

```
DROP USER IF EXISTS 'nombre_usuario'@'localhost';
```

Referencia:

- Documentación oficial de DROP USER en MySQL

### 4.9.2 Crear un nuevo usuario

La sintaxis simplificada para crear un usuario en MySQL es la siguiente:

```
CREATE USER [IF NOT EXISTS]
    user [auth_option] [, user [auth_option]] ...
    DEFAULT ROLE role [, role ] ...
    [REQUIRE {NONE | tls_option [[AND] tls_option] ...}]
    [WITH resource_option [resource_option] ...]
    [password_option | lock_option] ...
    [COMMENT 'comment_string' | ATTRIBUTE 'json_object']
```

**Ejemplo:**

Habrá que reemplazar *nombre_usuario* y *contraseña* por los datos del nuevo usuario que desea crear:

```
CREATE USER 'nombre_usuario'@'localhost' IDENTIFIED BY 'contraseña';
```

Una vez que hemos creado el usuario hay que asignarle permisos para que pueda acceder a la/s base/s de datos que queramos.

Referencia:

- Documentación oficial de `CREATE USER` en MySQL

### 4.9.3 Tipos de permisos que podemos aplicar

- `ALL PRIVILEGES`: permite a un usuario de MySQL acceder a todas las bases de datos asignadas en el sistema.
- `CREATE`: permite crear nuevas tablas o bases de datos.
- `DROP`: permite eliminar tablas o bases de datos
- `DELETE`: permite eliminar registros de tablas.
- `INSERT`: permite insertar registros en tablas.
- `SELECT`: permite leer registros en las tablas.
- `UPDATE`: permite actualizar registros seleccionados en tablas.
- `GRANT OPTION`: permite asignar privilegios a otros usuarios.

### 4.9.4 Asignar permisos a un usuario

La sintaxis simplificada para asignar permisos a un usuario en MySQL es la siguiente:

```
GRANT [permiso] ON [nombre_base_de_datos].[nombre_tabla] TO 'nombre_usuario'@'
    localhost';
```

**Ejemplo:**

```
GRANT ALL PRIVILEGES ON *.* TO 'nombre_usuario'@'localhost';
```

En este comando, los asteriscos indican que estamos aplicando el permiso `ALL PRIVILEGES` al usuario `nombre_usuario` para todas las tablas de cada una de las bases de datos.

Después de este comando habrá que ejecutar el siguiente comando para refrescar todos los privilegios a los usuarios.

```
FLUSH PRIVILEGES;
```

Referencia:

- Documentación oficial de GRANT en MySQL

### 4.9.5 Eliminar permisos a un usuario

La sintaxis simplificada para elminar permisos a un usuario en MySQL es la siguiente:

```
REVOKE [permiso] ON [nombre_base_de_datos].[nombre_tabla] FROM 'nombre_usuario'
    @'localhost';
```

Referencia:

- Documentación oficial de REVOKE en MySQL

### 4.9.6 Consultar los usuarios creados en MySQL Server

Los usuarios de MySQL se almacenan en la tabla `mysql.user`. La clave primaria de esta tabla está formada por los valores `user` y `host`, de modo que cada fila vendrá identificada por un nombre de usuario y el host desde el que puede conectarse.

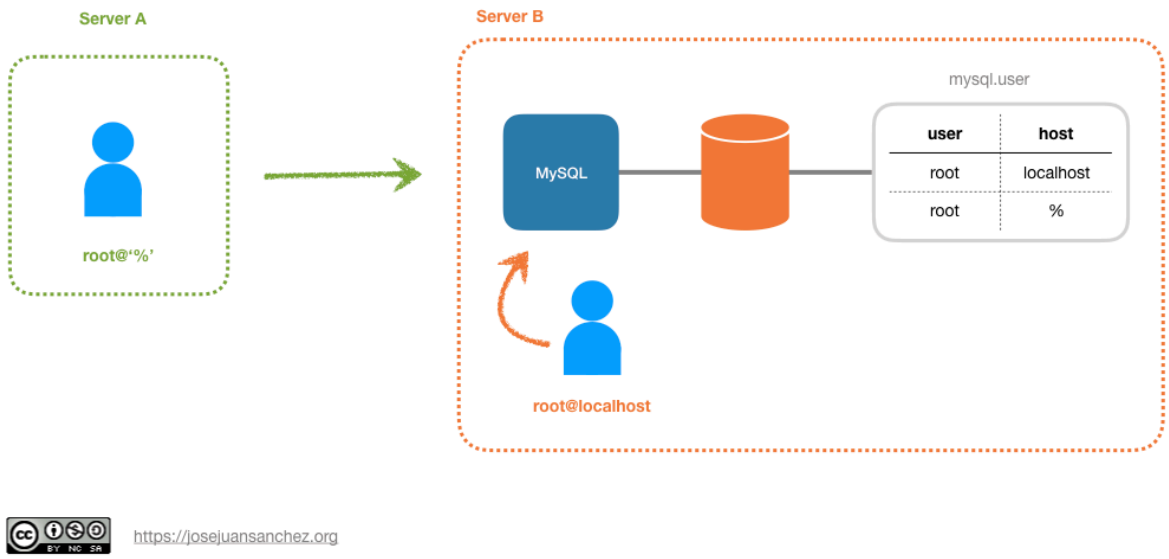
La siguiente consulta nos devuelve el listado de usuarios que tenemos en MySQL y desde qué host pueden conectarse:

```
SELECT user,host FROM mysql.user;
```

En nuestra caso la consulta anterior devuelve el siguiente resultado:

```
+------------------+--------------+
| user             | host         |
+------------------+--------------+
| root             | localhost    |
| debian-sys-maint | localhost    |
| mysql.session    | localhost    |
| mysql.sys        | localhost    |
+------------------+--------------+
```

El siguiente diagrama muestra un ejemplo de dos usuarios que se están conectando a una máquina con MySQL Server. El usuario `root@localhost` es un usuario que sólo puede conectarse desde la máquina local y el usuario `root@'%'` es un usuario que se puede conectar desde una máquina remota.




También podemos consultar qué permisos específicos tiene un determinado usuario. La siguiente consulta nos devuelve los permisos que tiene el usuario `root`:

```
SHOW GRANTS FOR root@localhost;
```

```
+----------------------------------------------------+
| Grants for root@localhost                          |
+----------------------------------------------------+
| GRANT ALL PRIVILEGES ON *.* TO 'root'@'localhost' |
+----------------------------------------------------+
```

## 4.10 Cómo ejecutar un script `.sql` desde la consola

(1) Desde la consola de Linux:

```
mysql -u root -p < script.sql
```

(2) Desde la consola de MySQL:

```
mysql> source script.sql
```

## 4.11 Cómo ejecutar sentencias SQL desde un script de bash

**Opción 1**

Utilizando el operador de redirección < con el comando `mysql`.

```
mysql -u $DB_USER -p$DB_PASSWORD $DB_NAME < script.sql
```

Donde:

- `$DB_USER`: es el nombre de usuario de la base de datos.
- `$DB_PASSWORD`: es la contraseña del usuario de la base de datos.
- `$DB_NAME`: es el nombre de la base de datos donde queremos ejecutar las sentencias.
- `script.sql`: es el nombre del archivo que contiene las sentencias SQL.

**Opción 2**

Utilizando el parámetro `-e` del comando `mysql`.

```
mysql -u $DB_USER -p$DB_PASSWORD $DB_NAME -e "SELECT * FROM tabla;"
```

**Opción 3**

Utilizando el operador de redirección <<< con el comando `mysql`.

El operador de redirección <<< permite utilizar una cadena como entrada estándar (*stdin*) a un comando.

Ejemplo:

```
#!/bin/bash

# Variables de configuración
DB_USER=usuario
DB_PASSWORD=contraseña
DB_NAME=base_de_datos

# Ejecutamos las sentencias SQL
mysql -u root <<< "DROP USER IF EXISTS '$DB_USER'@'%'"
mysql -u root <<< "CREATE USER '$DB_USER'@'%' IDENTIFIED BY '$DB_PASSWORD'"
mysql -u root <<< "GRANT ALL PRIVILEGES ON $DB_NAME.* TO '$DB_USER'@'%'"
```

Referencia:

- Redirection in bash

# 5 PHP

## 5.1 Instalación de módulos PHP

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

Para averiguar lo que hace el módulo `libapache2-mod-php`, podríamos escribir esto:

```
sudo apt show libapache2-mod-php
```

A continuación se describe brevemente qué es lo que contiene cada uno de ellos:

- `libapache2-mod-php`: Permite servir páginas PHP desde el servidor web apache.
- `php-mysql`: Permite conectar a una base de datos MySQL desde código PHP.

## 5.2 Comprobar que la instalación se ha realizado correctamente

Crea un archivo llamado `info.php` en el directorio `/var/www/html`.

```
sudo nano /var/www/html/info.php
```

Añade el siguiente contenido:

```
<?php

phpinfo();

?>
```

Ahora accede desde un navegador a la URL: `http://IP/info.php`, donde IP será la dirección IP de su máquina virtual. Por ejemplo, si la dirección IP de su máquina virtual es `192.168.22.200`, la URL será: `http://192.168.22.200/info.php`

# 6 Otras herramientas relacionadas con la pila LAMP

## 6.1 Instalar phpMyAdmin para acceder vía web a MySQL

### 6.1.1 Instalación de phpMyAdmin con `apt`

**Paso 1**

Para instalar `phpMyAdmin` necesitamos instalar los siguientes paquetes:

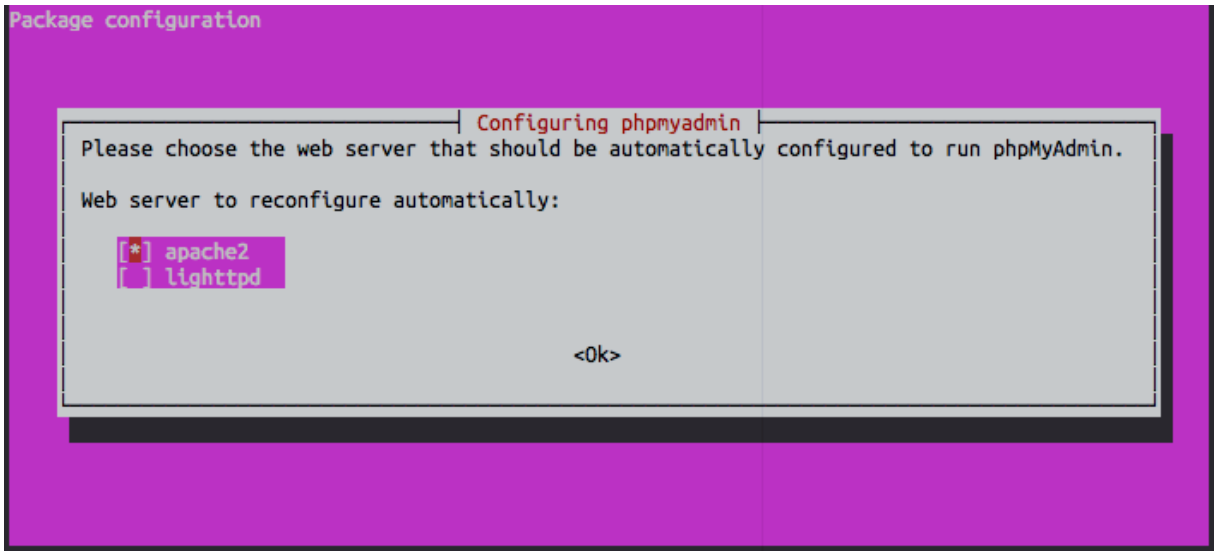
```
sudo apt install phpmyadmin php-mbstring php-zip php-gd php-json php-curl -y
```

A continuación se describe brevemente qué es lo que contiene cada uno de ellos:

- `php-mbstring`: Incluye el módulo mbstring (multi-byte string) que permite administrar cadenas no-ASCII y convertir cadenas a diferentes codificaciones.
- `php-zip`: Permite la carga de archivos .zip a `phpMyAdmin`.
- `php-gd`: Incluye la librería GD Graphics que permite crear y modificar imágenes.
- `php-json`: Añade soporte para trabajar con el formato JSON desde PHP.
- `php-curl`: Permite interactuar con servidores haciendo uso de diferentes protocolos desde PHP.

**Paso 2**

**Importante:** Durante el proceso de instalación le aparecerá una ventana para preguntarle qué servidor web quiere configurar para ejecutar phpMyAdmin. En nuestro caso seleccionaremos el servidor web **apache2**. Para seleccionarlo tiene que pulsar la barra espaciadora.




**Paso 3**

Confirme que desea utilizar `dbconfig-common` para configurar la base de datos.

**Paso 4**

Finalmente se le solicitará una contraseña para phpMyAdmin.

Una vez finalizado el proceso de instalación podrá acceder a la interfaz web de `phpMyAdmin` desde la URL: `http://IP/phpmyadmin`, donde IP será la dirección IP de su máquina virtual. Por ejemplo, si la dirección IP de su máquina virtual es `192.168.22.200`, la URL será: `http://192.168.22.200/phpmyadmin`

**¿Cómo podríamos automatizar esta instalación desde un script de Bash?**

Si quisiéramos automatizar el proceso de instalación de `phpMyAdmin` con `apt` tendríamos que hacer uso de la herramienta `debconf-set-selections`.

Las distribuciones Linux basadas en Debian utilizan la utilidad `debconf` para hacer preguntas al usuario durante el proceso de instalación de un paquete.

La utilidad `debconf-set-selections` nos permite automatizar las respuestas que dará el usuario durante del proceso de instalación. Esta utilidad es muy útil para poder automatizar la instalación de paquetes en un script de Bash sin intervención del usuario.

En el caso de `phpmyadmin` las respuestas que queremos automatizar son las siguientes:

(1) Seleccionar el servidor web que queremos configurar para ejecutar.

```
echo "phpmyadmin phpmyadmin/reconfigure-webserver multiselect apache2" |
    debconf-set-selections
```

(2) Confirmar que desea utilizar `dbconfig-common` para configurar la base de datos.

```
echo "phpmyadmin phpmyadmin/dbconfig-install boolean true" | debconf-set-
    selections
```

(3) Seleccionar la contraseña para phpMyAdmin.

```
echo "phpmyadmin phpmyadmin/mysql/app-pass password $PHPMYADMIN_APP_PASSWORD" |
    debconf-set-selections
echo "phpmyadmin phpmyadmin/app-password-confirm password
   $PHPMYADMIN_APP_PASSWORD" | debconf-set-selections
```

En este ejemplo estamos guardando la contraseña en la variable de entorno `$PHPMYADMIN_APP_PASSWORD` que tendremos que configurar previamente asignándole un valor inicial.

### 6.1.2 Otros métodos para instalar phpMyAdmin

En la documentación oficial de phpMyAdmin, encontramos todas las posibilidades que tenemos para realizar la instalación de phpMyAdmin.

- Instalación desde Git.
- Instalación usando Composer.
- Instalación usando Docker.
- Instalación rápida.

## 6.2 Instalar Adminer para acceder vía web a MySQL

Adminer es una alternativa a phpMyAdmin. Tiene la ventaja que se distribuye en un único archivo .php. Sólo hay que descargarse el archivo Adminer para MySQL que está disponible en la web del proyecto y guardarlo en el directorio `/var/www/html`.

## 6.3 Instalar un analizador de logs para Apache Server

Investiga cuáles son los analizadores de logs que existen para Apache Server e instala uno de ellos.

Posibles soluciones:

- GoAccess
- AWStats

### 6.3.1 GoAccess

Para instalar la última versión de GoAccess añadimos los repositorios oficiales del proyecto:

#### 6.3.1.1 Instalación de GoAccess

Actualizamos los repositorios e instalamos GoAccess.

```
sudo apt update
sudo apt install goaccess -y
```

#### 6.3.1.2 Uso de GoAccess

Una vez que hemos instalado la utilidad podemos usarla para procesar los archivos de log `access.log` de Apache HTTP Server.

**6.3.1.2.1 Desde el terminal** El siguiente comando parsea el archivo de log `access.log` y muestra la información del log en tiempo real en el terminal en modo texto.

```
goaccess /var/log/apache2/access.log -c
```

**6.3.1.2.2 Creación de un archivo HTML estático** El siguiente comando parsea el archivo de log `access.log` y genera un archivo HTML estático.

```
goaccess /var/log/apache2/access.log -o /var/www/html/report.html --log-format=
    COMBINED
```

**6.3.1.2.3 Creación de un archivo HTML en tiempo real** El siguiente comando parsea el archivo de log `access.log` y genera un archivo HTML en tiempo real.

```
goaccess /var/log/apache2/access.log -o /var/www/html/report.html --log-format=
    COMBINED --real-time-html
```

**Nota:** Si queremos generar el archivo HTML en tiempo real es necesario abrir el puerto **7890** en el firewall, porque es el puerto que utiliza por defecto el WebSocket del cliente JavaScript para comunicarse con la aplicación `goaccess`.

#### 6.3.1.3 Creación de un archivo HTML en tiempo real en segundo plano

Para ejecutar `goaccess` en segundo plano podemos utilizar la opción `--daemonize`.

```
goaccess /var/log/apache2/access.log -o /var/www/html/report.html --log-format=
    COMBINED --real-time-html --daemonize
```

## 6.4 Control de acceso a un directorio con autenticación básica

Crea un directorio llamado `stats` dentro del directorio `/var/www/html` donde se podrán consultar los informes generados con `goaccess`. El acceso a este directorio deberá estar controlado y solo se podrá acceder mediante un usuario y una contraseña.

**Paso 1**

Creamos el directorio `stats`.

```
mkdir -p /var/www/html/stats
```

**Paso 2**

Lanzamos el proceso de `goaccess` en background para generar los informes en segundo plano. Para ejecutar `goaccess` en segundo plano podemos utilizar el parámetro `--daemonize` o ejecutarlo con las utilidades `nohup`, `screen` o `tmux`.

Veamos cómo sería la **primera solución (recomendada)**, haciendo uso del parámetro `--daemonize`.

```
goaccess /var/log/apache2/access.log -o /var/www/html/stats/index.html --log-format=COMBINED --daemonize
```

Veamos cómo sería la **segunda solución**, haciendo uso de las utilidades `nohup`, `screen` o `tmux`.

```
sudo goaccess /var/log/apache2/access.log -o /var/www/html/stats/index.html --log-format=COMBINED --real-time-html &
```

**Nota:** Tiene que tener en cuenta que si ha iniciado el proceso en segundo plano con el carácter ampersand (&) al final del comando, el proceso finalizará cuando se cierre la sesión SSH.

Para evitar que el proceso finalice la sessión SSH podemos iniciarlo con alguna de las siguientes utilidades: `nohup`, `screen` o `tmux`.

- `nohup`: Esta utilidad ejecuta el comando que recibe como parámetro y hace que ignore las señales `SIGHUP`, que son las señales que se envían a un proceso cuando el terminal que los controla se cierra.

Ejemplo:

```
nohup goaccess /var/log/apache2/access.log -o /var/www/html/stats/index.html --log-format=COMBINED --real-time-html &
```

- `screen`: Esta utilidad permite iniciar una sesión y tener varios terminales virtuales en ella. Los procesos que se ejecutan en estos terminales virtuales no finalizarán al cerrar el terminal virtual.

Ejemplo:

```
screen -dmL goaccess /var/log/apache2/access.log -o /var/www/html/stats/index.html --log-format=COMBINED --real-time-html
```

Las opciones que se han utilizado en el ejemplo anterior son: `-d` inicia una sesión y automáticamente se desconecta de ella, `-m` fuerza la creación de una nueva sesión y `-L` habilita el log.

- `tmux`: Es una alternativa más reciente que `screen`. Realiza la misma funcionalidad que `screen`.

**Paso 3**

Creamos el archivo de contraseñas para el usuario que accederá al directorio `stats`. El archivo de contraseñas lo guardamos en un directorio seguro que no sea accesible desde el exterior. En nuestro caso el archivo se llamará `.htpasswd` y lo vamos a guardar en el directorio `/etc/apache2`. El usuario que vamos a crear tiene como nombre de usuario: `usuario`.

```
sudo htpasswd -c /etc/apache2/.htpasswd usuario
```

Este comando nos va a preguntar por la contraseña del usuario que estamos creando y por lo tanto no podríamos utilizarlo en un script de bash. Para poder incluir este comando en un script de bash tenemos que utilizar la opción `-b`, de este modo podemos indicarle el nombre del usuario y la contraseña como parámetros.

```
sudo htpasswd -bc /etc/apache2/.htpasswd $STATS_USERNAME $STATS_PASSWORD
```

Donde las variables `$STATS_USERNAME` y `$STATS_PASSWORD` estarán definidas previamente en el script de bash.

**Paso 4**

Editamos el archivo configuración de Apache.

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

Añadimos la siguiente sección dentro de las etiquetas de `<VirtualHost *:80>` y `</VirtualHost>`.

```
<Directory "/var/www/html/stats">
  AuthType Basic
  AuthName "Acceso restringido"
  AuthBasicProvider file
  AuthUserFile "/etc/apache2/.htpasswd"
  Require valid-user
</Directory>
```

De modo que el archivo de configuración quedará de forma similar al siguiente ejemplo.

```
<VirtualHost *:80>
        #ServerName www.example.com
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html

        <Directory "/var/www/html/stats">
          AuthType Basic
          AuthName "Acceso restringido"
          AuthBasicProvider file
          AuthUserFile "/etc/apache2/.htpasswd"
          Require valid-user
        </Directory>

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

A continuación se describe en qué consiste cada una de las directivas:

- `AuthType Basic`: Esta directiva indica que se va a utilizar el tipo de autenticación más básico, las opciones disponibles son `None`, `Basic`, `Digest` y `Form`. Este tipo de autenticación utiliza una codificación en Base64 y sólo se recomienda su uso en conexiones cifradas por SSL.
- `AuthName`: Esta directiva nos permite condigurar la cadena de texto que le aparecerá al usuario en el cuadro de diálogo.

- `AuthBasicProvider file`: En este caso es opcional, porque la opción `file` es el valor por defecto para esta directiva. Sólo será necesario utilizar esta directiva en caso de utilizar otro medio diferente para la autenticación.
- `AuthUserFile`: Indica la ruta donde se encuentra el archivo de contraseñas que hemos generado.
- `Require`: Esta directiva nos permite indicar el proveedor de autenticación que vamos a utilizar. Con la opción `Require user userid [userid] ...` podemos indicar la lista de usuarios a los que se le permite el acceso. También es posible indicar los nombres de los grupos de usuarios a los que se les permite el acceso con la opción `Require group group-name [group-name] ...`. La opción `Require valid-user` permite el acceso a cualquier usuario que aparezca en el archivo de contraseñas.

Puede encontrar más información en la documentación oficial de Apache HTTP Server.

**Paso 5**

Reiniciamos el servicio de Apache.

```
sudo systemctl restart apache2
```

## 6.5 Control de acceso a un directorio con `.htaccess`

En esta sección se explica cómo realizar el control de acceso a un directorio utilizando archivos `.htaccess`.

Los archivos `.htaccess` permiten realizar cambios en la configuración del servidor web Apache sin tener que modificar los archivos principales de configuración. Los archivos `.htaccess` contienen las directivas de configuración que queremos aplicar sobre un directorio específico y todos sus subdirectorios.

En la documentación oficial de Apache HTTP Server puede encontrar más información sobre los archivos `.htaccess`.

**Paso 1**

Creamos el directorio `stats`.

```
mkdir -p /var/www/html/stats
```

**Paso 2**

Lanzamos el proceso de `goaccess` en background para generar los informes en segundo plano.

```
sudo goaccess /var/log/apache2/access.log -o /var/www/html/stats/index.html --
    log-format=COMBINED --real-time-html &
```

**Paso 3**

Creamos el archivo de contraseñas para el usuario que accederá al directorio `stats`. El archivo de contraseñas lo guardamos en un directorio seguro que no sea accesible desde el exterior. En nuestro caso el archivo se llamará `.htpasswd` y lo vamos a guardar en el directorio `/etc/apache2`. El usuario que vamos a crear tiene como nombre de usuario: `usuario`.

```
sudo htpasswd -c /etc/apache2/.htpasswd usuario
```

Este comando nos va a preguntar por la contraseña del usuario que estamos creando y por lo tanto no podríamos utilizarlo en un script de bash. Para poder incluir este comando en un script de bash tenemos que utilizar la opción `-b`, de este modo podemos indicarle el nombre del usuario y la contraseña como parámetros.

```
sudo htpasswd -bc /etc/apache2/.htpasswd $STATS_USERNAME $STATS_PASSWORD
```

Donde las variables `$STATS_USERNAME` y `$STATS_PASSWORD` estarán definidas previamente en el script de bash.

**Paso 4**

Creamos el archivo `.htaccess` dentro del directorio que queremos proteger con usuario y contraseña. En nuestro caso lo vamos a crear en el directorio `/var/www/html/stats/.htaccess`.

```
sudo nano /var/www/html/stats/.htaccess
```

El contenido del archivo `.htaccess` será el siguiente:

```
AuthType Basic
AuthName "Acceso restringido"
AuthBasicProvider file
AuthUserFile "/etc/apache2/.htpasswd"
Require valid-user
```

A continuación se describe en qué consiste cada una de las directivas:

- `AuthType Basic`: Esta directiva indica que se va a utilizar el tipo de autenticación más básico, las opciones disponibles son `None`, `Basic`, `Digest` y `Form`. Este tipo de autenticación utiliza una codificación en Base64 y sólo se recomienda su uso en conexiones cifradas por SSL.
- `AuthName`: Esta directiva nos permite condigurar la cadena de texto que le aparecerá al usuario en el cuadro de diálogo.
- `AuthBasicProvider file`: En este caso es opcional, porque la opción `file` es el valor por defecto para esta directiva. Sólo será necesario utilizar esta directiva en caso de utilizar otro medio diferente para la autenticación.
- `AuthUserFile`: Indica la ruta donde se encuentra el archivo de contraseñas que hemos generado.
- `Require`: Esta directiva nos permite indicar el proveedor de autenticación que vamos a utilizar. Con la opción `Require user userid [userid] ...` podemos indicar la lista de usuarios a los que se le permite el acceso. También es posible indicar los nombres de los grupos de usuarios a los que se les permite el acceso con la opción `Require group group-name [group-name] ...`. La opción `Require valid-user` permite el acceso a cualquier usuario que aparezca en el archivo de contraseñas.

Puede encontrar más información en la documentación oficial de Apache HTTP Server.

**Paso 5**

Editamos el archivo configuración de Apache.

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

Añadimos la siguiente sección dentro de las etiquetas de `<VirtualHost *:80>` y `</VirtualHost>`.

```
<Directory "/var/www/html/stats">
  AllowOverride All
</Directory>
```

De modo que el archivo de configuración quedará de forma similar al siguiente ejemplo.

```
<VirtualHost *:80>
  #ServerName www.example.com
  ServerAdmin webmaster@localhost
  DocumentRoot /var/www/html

  <Directory "/var/www/html/stats">
    AllowOverride All
  </Directory>

  ErrorLog ${APACHE_LOG_DIR}/error.log
  CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

**Paso 6**

Reiniciamos el servicio de Apache.

```
sudo systemctl restart apache2
```

# 7 Referencias


- Cómo instalar en Ubuntu 18.04 la pila LAMP — Linux, Apache, MySQL y PHP
- «Apache: Soluciones y ejemplos para administradores de Apache». **Ken Coar y Rich Bowen**. O'Reilly.
- Aulas en red, aplicaciones y servicios. Linux. INTEF.
- Crear un nuevo usuario y otorgarle permisos en MySQL
- Instalación de phpMyAdmin.
- Documentación oficial de Apache HTTP Server.
- Autenticación y Autorización en Apache HTTP Server.
- How to Reset the Root Password. Documentación oficial de MySQL.
- How to reset root MySQL/MariaDB password on Ubuntu 20.04 Focal Fossa Linux.
- Cómo instalar y proteger phpMyAdmin en Ubuntu 20.04.

# 8 Licencia