
LEMP Stack

Implantación de Aplicaciones Web

Curso 2025/2026

Índice general

1	LEMP Stack	1
1.1	Instalación del servidor web Nginx	1
1.2	Instalación de php-fpm y php-mysql	1
1.2.1	php-fpm	1
1.2.2	php-mysql	1
1.3	Configuración de Nginx para comunicarse con php-fpm a través de un socket UNIX	2
1.4	Comprobar que la instalación se ha realizado correctamente	3
1.5	Configuración de Nginx para comunicarse con php-fpm a través de un socket TCP/IP	4
1.5.1	Opción 1: Nginx y php-fpm se ejecutan en la misma máquina	4
1.5.1.1	Configuración de php-fpm	4
1.5.1.2	Configuración de Nginx	5
1.5.2	Opción 2: Nginx y php-fpm se ejecutan diferentes máquinas	7
1.6	Configuración de la directiva <code>cgi.fix_pathinfo</code> para mejorar la seguridad	7
2	Referencias	9
3	Licencia	10

1 LEMP Stack

En esta práctica vamos a instalar la pila **LEMP** que es una variación de la pila **LAMP**. La única diferencia es que usa el servidor **Nginx** en lugar de **Apache**.

Nginx está considerado como un servidor web ligero de alto rendimiento que además suele ser utilizado como **proxy inverso** y **balanceador de carga**.

1.1. Instalación del servidor web Nginx

```
1 sudo apt update
2 sudo apt install nginx -y
```

1.2. Instalación de php-fpm y php-mysql

1.2.1. php-fpm

El paquete **php-fpm** (**PHP FastCGI Process Manager**) es una implementación alternativa al *PHP FastCGI* con algunas **características adicionales útiles para sitios web con mucho tráfico**.

El uso de **PHP-FPM** (**PHP FastCGI Process Manager**) con **Nginx** es ideal porque **permite mejorar el consumo de memoria del servidor**, haciendo que el servidor tenga un bajo consumo de recursos, mejorando de esta manera su rendimiento y escalabilidad. **PHP-FPM se ejecutará como un servicio independiente de Nginx** que se va a encargar de interpretar las peticiones que incluyen código PHP que reciba el servidor **Nginx**. Cuando el servidor **Nginx** reciba una petición HTTP donde haya que procesar código PHP, el servidor **Nginx se comunicará con PHP-FPM a través de un socket UNIX o un socket TCP/IP** para recibir la respuesta del código PHP interpretado y servirla al cliente que realizó la petición.

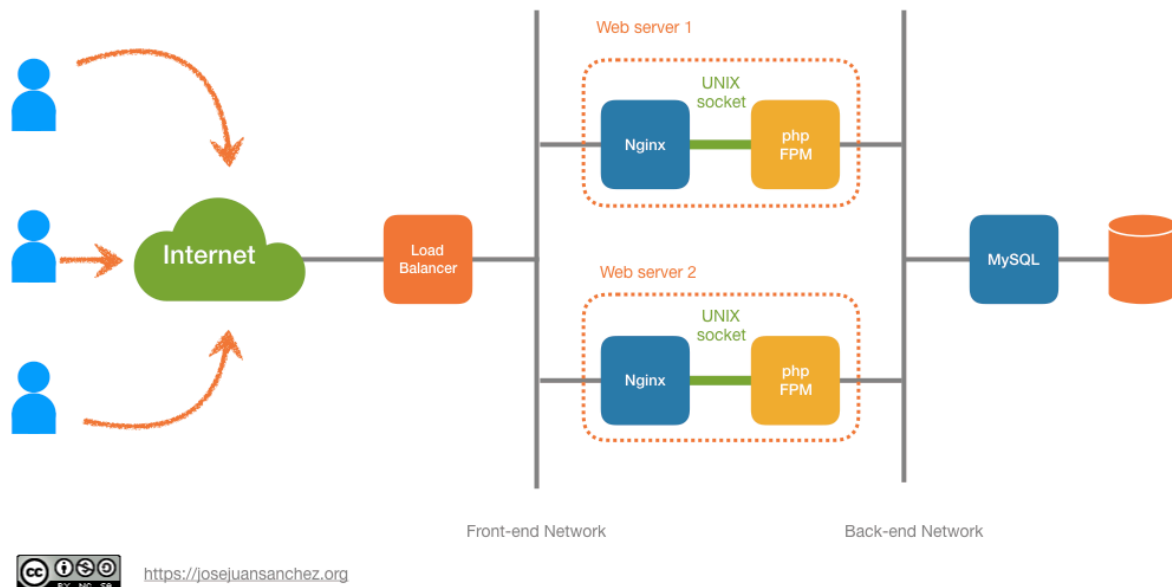
```
1 sudo apt install php-fpm -y
```

1.2.2. php-mysql

El paquete **php-mysql** permite a PHP interactuar con el sistema gestor de bases de datos MySQL.

```
1 sudo apt install php-mysql -y
```

1.3. Configuración de Nginx para comunicarse con php-fpm a través de un socket UNIX



Los **sockets UNIX** nos permiten realizar comunicación entre procesos, también conocida como **IPC (Inter-Process Communication)**, que es una función básica de los sistemas operativos que permite el intercambio de datos entre procesos de una forma eficiente. Sin embargo, los **sockets TCP/IP** nos permiten comunicar procesos a través de una red.

Un **sockets UNIX** es un tipo de archivo especial, donde los procesos pueden escribir y leer datos para comunicarse.

Los **sockets UNIX** tienen la **ventaja** que permiten realizar comunicaciones más rápidas entre los procesos, pero tienen el **inconveniente** de que son menos escalables que los **sockets TCP/IP** porque sólo permiten comunicar procesos que se están ejecutando en el mismo sistema operativo de la misma máquina.

En esta sección vamos a explicar cómo podemos configurar Nginx para que pueda comunicarse con el proceso **php-fpm** a través de un socket UNIX.

Editamos el archivo de configuración `/etc/nginx/sites-available/default`:

```
1 sudo nano /etc/nginx/sites-available/default
```

Realizamos los siguientes cambios:

- En la sección `index` añadimos el valor `index .php` en primer lugar para que darle prioridad respecto a los archivos `index.html`.
- Añadimos el bloque `location ~ \.php$` indicando dónde se encuentra el archivo de configuración `fastcgi-php.conf` y el archivo `php8.1-fpm.sock`.

- Opcionalmente podemos añadir el bloque `location ~ /\.ht` para no permitir que un usuario pueda descargar los archivos `.htaccess`. Estos archivos no son procesados por `Nginx`, son específicos de `Apache`.

Un posible archivo de configuración para el servidor podría ser el siguiente:

```
1 server {
2     listen 80 default_server;
3     listen [::]:80 default_server;
4
5     root /var/www/html;
6
7     index index.php index.html index.htm index.nginx-debian.html;
8
9     server_name _;
10
11    location / {
12        # First attempt to serve request as file, then
13        # as directory, then fall back to displaying a 404.
14        try_files $uri $uri/ =404;
15    }
16
17    # pass PHP scripts to FastCGI server
18    #
19    location ~ /\.php$ {
20        include snippets/fastcgi-php.conf;
21        # With php-fpm (or other unix sockets):
22        fastcgi_pass unix:/run/php/php8.1-fpm.sock;
23    }
24
25    # deny access to .htaccess files, if Apache's document root
26    # concurs with nginx's one
27    location ~ /\.ht {
28        deny all;
29    }
30 }
```

Nota: En el momento de redactar esta guía la versión de PHP es la 8.1. Tenga en cuenta que esta versión puede cambiar en un futuro.

Podemos comprobar que la sintaxis del archivo de configuración es correcta con el comando:

```
1 sudo nginx -t
```

Una vez realizados los cambios reiniciamos el servicio `nginx`:

```
1 sudo systemctl restart nginx
```

1.4. Comprobar que la instalación se ha realizado correctamente

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

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

Añade el siguiente contenido:

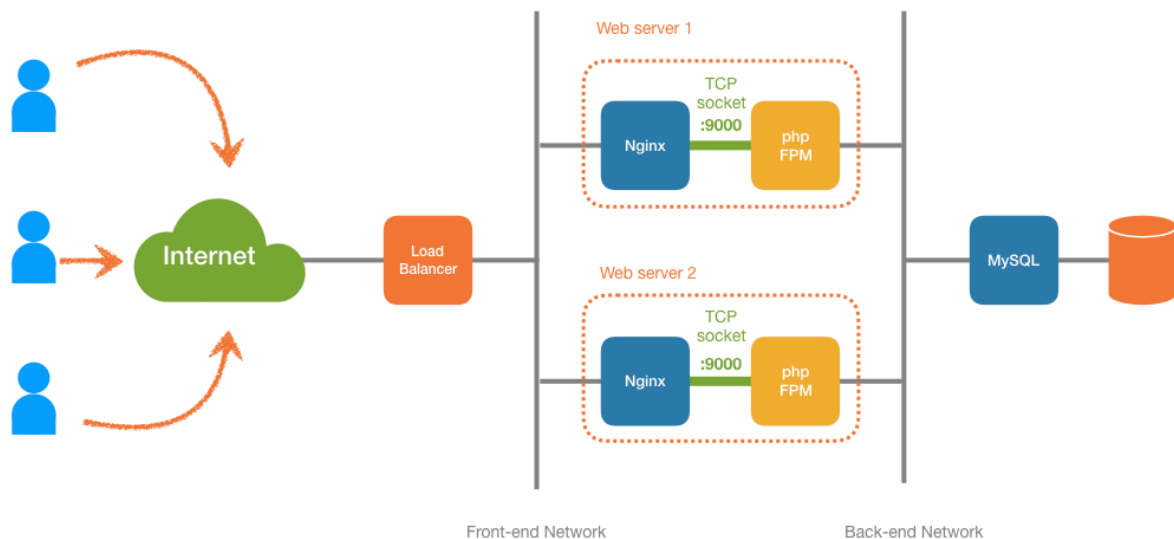
```
1 <?php
2
3 phpinfo();
4
5 ?>
```

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`

1.5. Configuración de Nginx para comunicarse con php-fpm a través de un socket TCP/IP

Los **sockets TCP/IP** nos permiten comunicar procesos que se pueden estar ejecutando en la misma máquina o en máquinas diferentes, a través de una red.

1.5.1. Opción 1: Nginx y php-fpm se ejecutan en la misma máquina



<https://josejuansanchez.org>

1.5.1.1. Configuración de php-fpm

En primer lugar hay que modificar la directiva `listen` del archivo `/etc/php/8.1/fpm/pool.d/www.conf`.

```
1 sudo nano /etc/php/8.1/fpm/pool.d/www.conf
```

Si buscamos la directiva `listen` en el archivo de configuración nos encontramos que en la configuración por defecto está escuchando en el socket UNIX `/run/php/php8.1-fpm.sock`. A continuación se muestra un fragmento del archivo de configuración por defecto que hace referencia a la directiva `listen`.

```
1 ; The address on which to accept FastCGI requests.
2 ; Valid syntaxes are:
3 ;   'ip.add.re.ss:port'   - to listen on a TCP socket to a specific IPv4
   address on
4 ;                           a specific port;
5 ;   '[ip:6:addr:ess]:port' - to listen on a TCP socket to a specific IPv6
   address on
6 ;                           a specific port;
7 ;   'port'                 - to listen on a TCP socket to all addresses
   (IPv6 and IPv4-mapped) on a specific port;
8 ;
9 ;   '/path/to/unix/socket' - to listen on a unix socket.
10 ; Note: This value is mandatory.
11 listen = /run/php/php8.1-fpm.sock
```

Habrá que modificar la directiva `listen` por la dirección de localhost (`127.0.0.1`) y un puerto. En este ejemplo utilizaremos el puerto `9000`. La directiva `listen` quedaría así:

```
1 listen = 127.0.0.1:9000
```

Una vez que hemos realizado las modificaciones en la configuración reiniciamos el servicio de `php-fpm` para que se apliquen los cambios:

```
1 sudo systemctl restart php8.1-fpm
```

1.5.1.2. Configuración de Nginx

En este caso hay que configurar en el archivo `/etc/nginx/sites-available/default` que los scripts PHP se van a enviar al servidor FastCGI a través de un socket TCP/IP.

```
1 sudo nano /etc/nginx/sites-available/default
```

Habrá que modificar la directiva de configuración `fastcgi_pass` para indicar la dirección y el puerto donde se encuentra el servidor FastCGI. Por ejemplo, si el servidor FastCGI se está ejecutando en la misma máquina (`127.0.0.1`), en el puerto `9000` habrá que asignarle el siguiente valor:

```
1 fastcgi_pass 127.0.0.1:9000;
```

Un posible archivo de configuración para el servidor podría ser el siguiente:

```
1 server {
2     listen 80 default_server;
3     listen [::]:80 default_server;
4
5     root /var/www/html;
6 }
```

```
7     index index.php index.html index.htm index.nginx-debian.html;
8
9     server_name _;
10
11    location / {
12        # First attempt to serve request as file, then
13        # as directory, then fall back to displaying a 404.
14        try_files $uri $uri/ =404;
15    }
16
17    # pass PHP scripts to FastCGI server
18    #
19    location ~ \.php$ {
20        include snippets/fastcgi-php.conf;
21        # With php-cgi (or other tcp sockets):
22        fastcgi_pass 127.0.0.1:9000;
23    }
24
25    # deny access to .htaccess files, if Apache's document root
26    # concurs with nginx's one
27    location ~ /\.ht {
28        deny all;
29    }
30 }
```

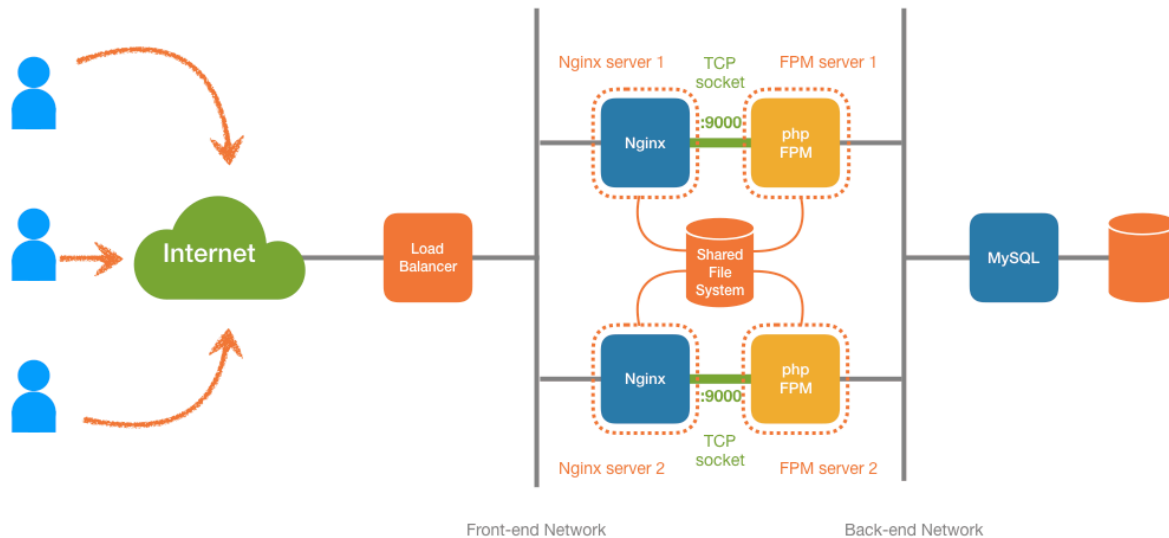
Podemos comprobar que la sintaxis del archivo de configuración es correcta con el comando:

```
1 sudo nginx -t
```

Una vez realizados los cambios reiniciamos el servicio `nginx`:

```
1 sudo systemctl restart nginx
```

1.5.2. Opción 2: Nginx y php-fpm se ejecutan diferentes máquinas



<https://josejuansanchez.org>

Al utilizar diferentes máquinas para el servidor web Nginx y php-fpm, debemos buscar una solución para compartir los archivos de contenido estático y el código fuente de la aplicación web, entre las dos máquinas.

Algunas de las soluciones que podemos utilizar para compartir el contenido entre los dos servidores son las siguientes:

- `rsync + cron`
- Sistema de archivos compartido por [NFS \(Network File System\)](#)
- Sistema de almacenamiento distribuido como [GlusterFS](#) o [Ceph](#) <!--
- Sistema de almacenamiento [NAS \(Network Attached Storage\)](#)
- [Storage Area Network \(SAN\)](#) ->

1.6. Configuración de la directiva `cgi.fix_pathinfo` para mejorar la seguridad

Es recomendable realizar un cambio en la directiva de configuración `cgi.fix_pathinfo` **por cuestiones de seguridad**. Editamos el siguiente archivo de configuración:

```
1 sudo nano /etc/php/8.1/fpm/php.ini
```

Nota: En el momento de redactar esta guía la versión de PHP es la 8.1. Tenga en cuenta que esta versión puede cambiar en un futuro.

Buscamos la directiva de configuración `cgi.fix_pathinfo` que por defecto aparece comentada con un punto y coma y con un valor igual a 1.

```
1 ;cgi.fix_pathinfo=1
```

Eliminamos el punto y coma y la configuramos con un valor igual a 0.

```
1 cgi.fix_pathinfo=0
```

Una vez modificado el archivo de configuración y guardados los cambios reiniciamos el servicio `php8.1-fpm`.

```
1 sudo systemctl restart php8.1-fpm
```

Referencias sobre los problemas de seguridad relacionados con la directiva `cgi.fix_pathinfo`:

- [Setting up PHP-FastCGI and nginx? Don't trust the tutorials: check your configuration!](#).
- [PHP FastCGI Example](#).

2 Referencias

- [Página oficial de Nginx.](#)
- [Getting started with Nginx.](#)
- [¿Cómo Instalar Linux, Nginx, MySQL, PHP \(LEMP stack\) en Ubuntu 20.04?](#)
- [Nginx vs Apache: Lucha entre Servidores Web.](#) Tonino Jankov.
- [How to Connect NGINX to PHP-FPM Using UNIX or TCP/IP Socket.](#) Aaron Kili.
- [Cómo crear un grupo de almacenamiento redundante con GlusterFS en Ubuntu 20.04.](#) Mark Drake.

3 Licencia

Esta página forma parte del curso Implantación de Aplicaciones Web de José Juan Sánchez Hernández y su contenido se distribuye bajo una licencia Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional.