# PHP

PHP es indispensable para un servidor con capacidades Web. Pero existen muchas variantes de como configurar PHP para ser usado en con un servidor web.

- PHP log cuando usamos PHP-FPM con host virtuales
- PHP. Como instalar una nueva versión de PHP en un sistema basado DEB
- Activar PHP8.2 JIT Compiler - Just-In-Time compilation (JIT)
- Instalar una versión de PHP con módulos en Ubuntu con Ondrej copiando de otra versión ya instalada

# PHP log cuando usamos PHP-FPM con host virtuales

# Introducción

Uno de esos tips que está infravalorado. Podremos ver, decenas de páginas, y cientos de respuestas, en las que todos hablan de cómo configurar el log, de si el path es incorrecto, o de si estas en linux o en windows. Al final el tema está en otro lugar, pues muchos de ellos, contestan si ni siquiera saber qué es PHP-FPM o cuando menos, sin verificar los que dicen que hace lo que escriben.

# Problema

En una instalación estándar de PHP-FPM en sistemas Linux, es posible que los errores de PHP no se registren correctamente en los logs de los dominios virtuales. Esto dificulta enormemente la depuración de problemas, ya que los mensajes de error críticos pueden perderse o no quedar registrados en los archivos de log esperados.

# Causa

Por defecto, PHP-FPM captura la salida estándar y de error de los workers (procesos de trabajo), pero no la redirige automáticamente a los logs. La directiva `catch_workers_output` está configurada como `no` en la configuración predeterminada, no existir o están simplemente comentada, lo que impide que los errores generados por los scripts PHP se registren adecuadamente.

# Solución

Para resolver este problema, es necesario modificar el archivo de configuración del pool de PHP-FPM:

1. Abrir el archivo de configuración del pool (normalmente ubicado en `/etc/php/X.X/fpm/pool.d/www.conf`, donde X.X es la versión de PHP, por ejemplo 7.4, 8.0, 8.1, etc.)
2. Buscar la directiva `catch_workers_output`
3. Cambiar su valor de `no` a `yes`:

   ```
   catch_workers_output = yes
   ```

4. Guardar el archivo y reiniciar el servicio PHP-FPM:

   ```
   sudo systemctl restart php7.4-fpm.service  # Ajustar según la versión de PHP
   ```

# Explicación técnica

Cuando `catch_workers_output = yes` está activado, PHP-FPM captura la salida de los procesos worker y la redirige adecuadamente al sistema de logs. Esto asegura que:

- Los errores de PHP aparezcan en los logs específicos del dominio virtual
- Los mensajes críticos no se pierdan
- Sea más fácil identificar y solucionar problemas en aplicaciones PHP

# Consideraciones adicionales

En versiones más recientes de PHP-FPM (7.3+), se recomienda usar las siguientes directivas en lugar de `catch_workers_output`:

```
php_admin_flag[log_errors] = on
php_admin_value[error_log] = /ruta/a/error.log
```

Estas opciones ofrecen un mejor rendimiento y mayor control sobre el destino de los logs.

# Importancia

Configurar correctamente los logs de error es un paso crucial para:

- Facilitar la depuración de aplicaciones
- Detectar problemas de seguridad
- Monitorizar el rendimiento del servidor
- Cumplir con buenas prácticas de administración de sistemas

Sin esta configuración, podrías estar operando tus aplicaciones PHP "a ciegas", sin visibilidad sobre los errores que ocurren.

# Otros enlaces

- directive 'catch_workers_output = yes' don't work as I want
- PHP-FPM doesn't write to error log

# Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

# PHP. Como instalar una nueva versión de PHP en un sistema basado DEB

# PHP. Como instalar una nueva versión de PHP en un sistema DEB basado

Lo habitual en mis máquinas es tener varias versiones de PHP usando el paquete de XXXXXX. Cuando sale una nueva versión suelo clonar la configuración de módulos, ya que a fin de cuentas luego será la nueva versión en los sitios cuando los actualice y verifique que pueden usar la nuevas version de PHP:

## Proceso de clonado de PHP a otra versión

### Obtener la lista de paquetes de la versión actual

Para obtener la lista de todos los paquetes de PHP 8.2 instalados en tu sistema Debian (Raspberry Pi), puedes usar dpkg con grep:

```
dpkg -l | grep php8.2 | awk '{print $2}' > php8.2-packages.txt
```

Esto creará un archivo php8.2-packages.txt que contiene todos los paquetes instalados relacionados con PHP 8.2.

# Crear un fichero con los nombres de paquete actualizados a la nueva versión

Ahora, necesitas cambiar 'php8.2' por 'php8.3' en todos los nombres de los paquetes para preparar la instalación de la nueva versión. Puedes hacerlo con sed:

```
sed 's/php8.2/php8.3/g' php8.2-packages.txt > php8.3-packages.txt
```

Esto generará un nuevo archivo php8.3-packages.txt con los nombres de los paquetes modificados para PHP 8.3.

# Script para instalar los paquetes de PHP 8.3

Ahora puedes crear un script que lea el archivo php8.3-packages.txt y use apt para instalar cada paquete. Aquí tienes un ejemplo de cómo podría ser este script:

```
#!/bin/bash

# Asegúrate de que el script se ejecute con privilegios de superusuario
if [ "$(id -u)" != "0" ]; then
   echo "Este script debe ser ejecutado como root" 1>&2
   exit 1
fi

# Actualiza los repositorios e instala los paquetes
apt update

while read package; do
    apt install -y $package
done < php8.3-packages.txt
```

Guarda este script como `upgrade_to_php8.3.sh`, por ejemplo.

# Ejecutar el script

Antes de ejecutar el script, asegúrate de que es ejecutable:

```
chmod +x upgrade_to_php8.3.sh
```

Luego, ejecútalo como root o utilizando sudo:

```
sudo ./upgrade_to_php8.3.sh
```

Este script actualizará tus paquetes a la versión 8.3 de PHP, asegurándose de mantener las mismas extensiones y configuraciones que tenías para la versión 8.2.

## Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

# Activar PHP8.2 JIT Compiler - Just-In-Time compilation (JIT)

## PHP8.2 JIT Compiler

Los lenguajes de programación puramente interpretados no tienen una fase de compilación y ejecutan directamente el código en una máquina virtual. La mayoría de los lenguajes interpretados, incluyendo PHP, de hecho, tienen una ligera fase de compilación para mejorar su rendimiento.

Por otro lado, los lenguajes de programación con compilación anticipada (AOT) requieren que el código sea compilado antes de su ejecución.

La compilación **Just-In-Time (JIT)** es un modelo híbrido entre el intérprete y la compilación anticipada, donde parte o todo el código se compila, a menudo en tiempo de ejecución, sin que el desarrollador tenga que compilarlo manualmente.

Históricamente, **PHP** fue un lenguaje interpretado, donde todo el código era interpretado por una máquina virtual **(Zend VM)**. Esto cambió con la introducción de Opcache y los **Opcodes**, que se generaban a partir del código PHP y podían ser almacenados en memoria caché. PHP 7.0 introdujo el concepto de **AST (Árbol de Sintaxis Abstracta**), que separó aún más el analizador sintáctico del compilador.

El JIT de PHP utiliza internamente **DynASM de LuaJIT** y está implementado como parte del **Opcache**.

El uso de JIT en PHP es altamente recomendable, no solo para propósito general, sino más aun, es aconsejable producir código de acuerdo a los cambios implementado desde PHP 8.0 que van mucho más allá de la activación de una forma de procesar en el servidor.

En particular lo uso en mis servidores en los que hago despliegues con Laravel y francamente, es impresionante.

## Activarlo en el servidor

En un servidor con PHP, en múltiples versiones, y con el modo FPM entre otros como uso con el servidor Web, y como ejemplo una Ubuntu, debemos activar JIT que por defecto no esta activado.

Encontraremos sus variables disponibles en el fichero `php.ini` de la versión en la que queremos activar JIT, pero como consejo, es mejor no hacerlo ahí.

En su lugar debemos hacerlo en `/etc/php/8.2/mods-available/` ya que es ahí donde están los ficheros parciales de cada modulo, y su configuración, tanto para PHP-FPM, como para **PHP en modo cli**.

> Esta es una buena práctica que existe en Ubuntu y otras distribuciones de Linux.

El fichero en cuestión es `/etc/php/8.2/mods-available/opcache.ini` y contiene por defecto. Esto no activará JIT sino que lo cargará pero lo mostrará **apagado**, por ejemplo usando `phpinfo()`

```
; configuration for php opcache module
; priority=10
zend_extension=opcache.so
```

> Si cambias los valores en el `php.ini` general es probable que todo este activado menos el par `JIT On`

JIT Disable

# Configuración ejemplo

Bien, os dejo una configuración bastante funcional en un sistema con uso muy intensivo tanto de php vía colas de Laravel usando php-cli y vía API externa en producción.

> Las mejoras son de 8,2 segundos en operaciones intensas a solo 1,2s.

```
zend_extension=opcache.so
opcache.jit=on
opcache.interned_strings_buffer=8
opcache.max_accelerated_files=10000
opcache.max_wasted_percentage=5
opcache.validate_timestamps=1
```

```
opcache.revalidate_freq=2
opcache.jit_buffer_size=100M
opcache.jit=tracing
opcache.jit=1255
opcache.save_comments=0
opcache.file_cache=/var/cache/php/opcache
```

Tras el reinicio de PHP-FPM verás algo parecido a esto:

JIT Activado

> Esta configuración `opcache.file_cache` requiere que exista `/var/cache/php/opcache` para servir de segundo nivel de cache (o el que tu definas), y que tenga los permisos para el ejecutor. En algunos sitios veo de usar `www-data` que es el que usa en una Ubuntu con Nginx. Puedes dejarlo vacío, si eres generoso con la memoriza asignada. En realidad, es algo así como el swap en linux.

Te aconsejo una vuelta por el fichero `php.ini` principal, para que veas todos los comentarios de cada uno delos valores. Ya sabes, el copia y pega, puede ser fatal, pues lo que a mi me funciona puede que a ti no.

# Enlaces

- Make your app faster with PHP 8.3

## Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.

# Instalar una versión de PHP con módulos en Ubuntu con Ondrej copiando de otra versión ya instalada

## Introducción

Un sistema rápido para clonar, por asi decirlo, la instalación actual de PHP en modo PHP-FPM con el repositorio Ondřej Surý

En este caso vamos a describir el instala la 8.0 teniendo una 8.2 instalada

## Paso 1: Obtener la Lista de Paquetes de PHP 8.2

1. **Listar los Paquetes de PHP 8.2 Instalados**:
   - Usa el siguiente comando para listar los paquetes de PHP 8.2 y guardarlos en un archivo de texto:

     ```
     dpkg -l | grep php8.2 | awk '{print $2}' > php8.2-packages.txt
     ```

   - Esto creará un archivo `php8.2-packages.txt` con los nombres de los paquetes instalados.

## Paso 2: Modificar la Lista para PHP 8.0

1. **Modificar el Archivo para Usar PHP 8.0**:
   - Usa `sed` para reemplazar `php8.2` por `php8.0` en el archivo:

```
sed 's/php8.2/php8.0/g' php8.2-packages.txt > php8.0-packages.txt
```

- Ahora tendrás un archivo `php8.0-packages.txt` con los nombres de los paquetes que necesitas instalar para PHP 8.0.

# Paso 3: Instalar los Paquetes de PHP 8.0

1. **Instalar los Paquetes Usando el Archivo**:
   - Usa el siguiente comando para instalar todos los paquetes listados en `php8.0-packages.txt`:

     ```
     sudo xargs -a php8.0-packages.txt apt-get install -y
     ```

   - Este comando usará `xargs` para pasar cada paquete listado en el archivo al comando `apt-get install`.

# Consideraciones Adicionales

- **Verificar la Instalación**: Después de la instalación, verifica que PHP 8.0 y sus módulos estén correctamente instalados y configurados.
- **Dependencias Adicionales**: Algunos módulos pueden requerir dependencias adicionales que no estén cubiertas por los paquetes listados. Asegúrate de revisar cualquier mensaje de error durante la instalación.

Siguiendo estos pasos, podrás automatizar el proceso de instalación de PHP 8.0 y sus módulos basándote en los paquetes de PHP 8.2 que ya tienes instalados.

## Aviso

Esta documentación y su contenido, no implica que funcione en tu caso o determinados casos. También implica que tienes conocimientos sobre lo que trata, y que en cualquier caso tienes copias de seguridad. El contenido el contenido se entrega, tal y como está, sin que ello implique ningún obligación ni responsabilidad por parte de Castris

Si necesitas soporte profesional puedes contratar con Castris soporte profesional.