

Cron en WordPress: cómo pasar de WP-Cron a cron real

Tareas a tiempo, incluso sin visitas · Guía práctica de 10 minutos

Por Fabini | [@fabinidc](#) · fabini.one · Guía gratuita

Conceptos rápidos: entiende el problema en 30 segundos

WP-Cron (predeterminado)

WordPress usa un sistema pseudo-cron que **se dispara con visitas**. Cada vez que alguien carga una página, WordPress comprueba si hay tareas pendientes. Parece práctico, pero tiene un problema fundamental: si nadie visita tu web, las tareas nunca se ejecutan.

Este sistema funciona bien en sitios con tráfico constante, pero falla estrepitosamente en blogs personales, páginas corporativas con pocas visitas o sitios en mantenimiento.

Cron real (sistema/hosting)

Un cron del servidor **se ejecuta por reloj del sistema**, independientemente del tráfico. Es el estándar Unix/Linux que lleva décadas funcionando en servidores de todo el mundo.

El cron real garantiza que tus tareas programadas se disparen exactamente cuando deben, sin depender de visitas aleatorias. Es la solución profesional que usan sitios empresariales y aplicaciones críticas.

Emails no salen

Los formularios de contacto, notificaciones automáticas o newsletters no se envían porque el sistema espera una visita para procesar la cola.

Posts programados no publican

Ese artículo que programaste para las 8:00 sigue en borrador a las 10:00 porque nadie ha visitado tu web desde entonces.

Backups fallan

Los plugins de backup no se ejecutan de forma fiable, dejando tu sitio vulnerable ante cualquier problema técnico o ataque.

Estos síntomas indican que dependes de WP-Cron basado en visitas. La solución es sencilla: desactivar WP-Cron y configurar un cron real del servidor que llame a wp-cron.php de forma programada y fiable.

Antes de empezar: requisitos y preparación

Esta guía asume que tienes conocimientos técnicos básicos de WordPress y acceso a los archivos de tu instalación. No es una operación complicada, pero requiere precisión para evitar problemas. Dedicar 10 minutos sin prisas y sigue cada paso con atención.



Backup reciente

Antes de tocar `wp-config.php` o crear tareas cron, asegúrate de tener un backup completo de tu base de datos y archivos. Usa tu plugin de backup habitual o descarga todo por FTP/SSH.



Acceso a archivos

Necesitas editar `wp-config.php` (en la raíz de WordPress) y acceder al panel de cron de tu hosting (cPanel, Plesk) o conectarte por SSH si gestionas tu propio servidor.



URL de `wp-cron.php`

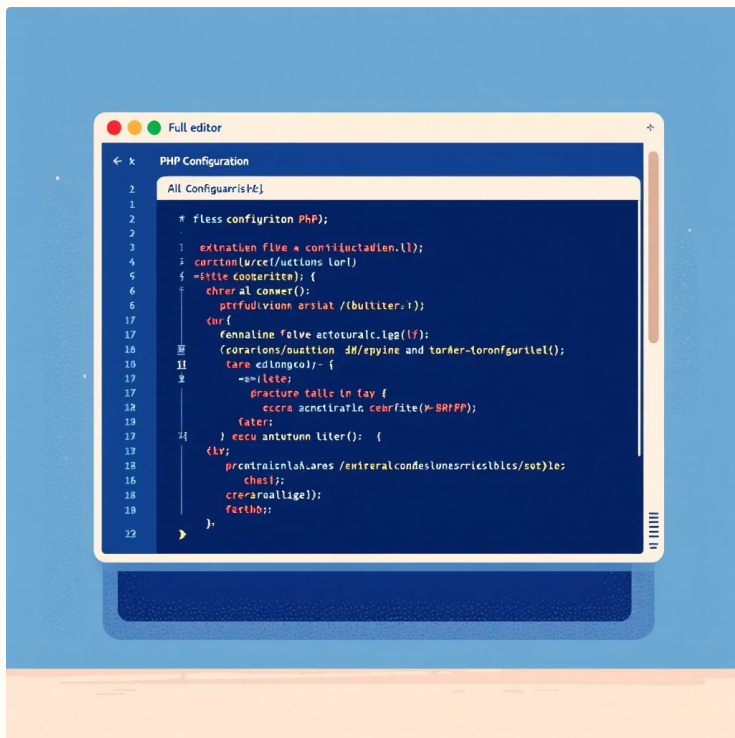
Localiza la URL completa de tu instalación. Será algo como `https://tu-dominio.com/wp-cron.php?doing_wp_cron`. Esta URL es la que invocarás desde el cron real.



Importante: Si utilizas un firewall de aplicación (WAF) o reglas `.htaccess` restrictivas, es posible que tengas que añadir excepciones para permitir que el servidor acceda a `wp-cron.php`. Apunta este detalle para revisarlo en el paso de troubleshooting si encuentras errores 403 o 404.

Una vez tengas estos tres elementos preparados, estarás listo para desactivar WP-Cron y configurar el cron real. El proceso es reversible: si algo falla, puedes eliminar la línea de `wp-config.php` y borrar la tarea cron para volver al comportamiento predeterminado.

Paso 1: Desactiva WP-Cron en WordPress



El primer paso es **desactivar el sistema WP-Cron basado en visitas**. Para ello, añadirás una constante de configuración al archivo wp-config.php que impide que WordPress lance tareas cron cuando un usuario carga una página.

Esta desactivación es segura y reversible. Simplemente elimina la línea si quieres volver al comportamiento anterior.

Edita wp-config.php

Localiza el archivo wp-config.php en la raíz de tu instalación WordPress (normalmente en /public_html/ o /htdocs/). Ábrelo con un editor de texto plano (Notepad++, Sublime Text, VSCode) o el editor de archivos de tu panel de hosting.

Busca la línea que dice `/* That's all, stop editing! Happy publishing. */` o cualquier comentario similar que indique el final de las configuraciones personalizadas. Justo antes de esa línea, añade el siguiente código:

```
/* wp-config.php — desactiva WP-Cron basado en visitas */
define('DISABLE_WP_CRON', true);
```

Guarda los cambios y sube el archivo a tu servidor si lo editaste en local. A partir de este momento, WordPress ya no ejecutará tareas cron automáticamente con las visitas.

 **Nota crítica:** Hasta que crees el cron real en el siguiente paso, tus tareas programadas no correrán automáticamente. Esto significa que emails, publicaciones programadas y backups quedarán en pausa. Por eso es importante completar el Paso 2 inmediatamente después de este cambio.

Para verificar que la constante está activa, puedes instalar temporalmente un plugin como WP Control y comprobar que muestra un aviso indicando que WP-Cron está deshabilitado. Esto confirma que el cambio se aplicó correctamente.

Paso 2A: Crea el cron real desde el panel de hosting

La mayoría de proveedores de hosting (Webempresa, Raiola, SiteGround, Hostinger, etc.) ofrecen un panel de administración (cPanel o Plesk) con una sección dedicada a tareas cron. Esta es **la opción más sencilla** si no tienes experiencia con la línea de comandos.

Accede a la sección de Cron Jobs

Desde tu panel de hosting, busca una opción llamada "Cron Jobs", "Tareas Cron", "Tareas programadas" o similar. En cPanel, suele estar en la sección "Avanzado". En Plesk, bajo "Tareas programadas".

01

Selecciona la frecuencia

Configura el intervalo de ejecución. Lo habitual es **cada 5 a 15 minutos**. Para un sitio con tráfico moderado, 10 minutos es un buen equilibrio entre puntualidad y carga del servidor.

02

Introduce el comando PHP

En el campo de comando, pega la ruta completa al binario de PHP y al archivo wp-cron.php de tu instalación. Ejemplo orientativo para usar el intérprete PHP:

03

Redirige la salida

Añade `> /dev/null 2>&1` al final del comando para evitar que el servidor te envíe emails con cada ejecución. Esto es opcional pero recomendado para no saturar tu bandeja de entrada.

```
/usr/bin/php -q /home/[[usuario]]/public_html/wp-cron.php > /dev/null 2>&1
```

Alternativa: usar cURL para invocar por HTTP

Si no conoces la ruta exacta del binario PHP o prefieres una solución más universal, puedes usar cURL para llamar a wp-cron.php por HTTP. Este método funciona igual de bien y es más fácil de configurar:

```
/usr/bin/curl -s https://[[tu-dominio.com]]/wp-cron.php?doing_wp_cron > /dev/null 2>&1
```

El parámetro `-s` hace que cURL trabaje en modo silencioso. La URL debe incluir `?doing_wp_cron` para que WordPress reconozca la petición como una ejecución legítima del cron.



Consejo de seguridad: Si usas el método HTTP (cURL), considera bloquear el acceso público a wp-cron.php mediante reglas de firewall o .htaccess, permitiendo solo las peticiones desde la IP del servidor. Esto previene que terceros disparen el cron manualmente y sobrecarguen tu sitio. Veremos cómo hacerlo en la sección de opciones avanzadas.

Paso 2B: Crea el cron real en Linux (crontab)

Si gestionas tu propio servidor VPS o dedicado (Ubuntu, Debian, CentOS, etc.), configurarás el cron directamente desde la línea de comandos usando `crontab`. Este método te da **control total** sobre la ejecución y es el estándar en entornos profesionales.

Edita el crontab del usuario web

Conéctate a tu servidor por SSH y ejecuta el siguiente comando para editar las tareas cron del usuario que ejecuta el servidor web (normalmente `www-data`, `apache` o tu usuario personal):

```
# Editar el crontab del usuario actual
crontab -e
```

Se abrirá un editor de texto (nano, vim o el que tengas configurado). Añade una nueva línea al final del archivo con la configuración del cron. La sintaxis cron usa cinco campos para definir cuándo ejecutar la tarea:

```
# Minuto Hora Día-mes Mes Día-semana Comando
*/10 * * * * /usr/bin/php -q /var/www/[[tu-sitio]]/public_html/wp-cron.php > /dev/null 2>&1
```

Esta línea ejecuta `wp-cron.php` **cada 10 minutos** (`*/10` en el campo de minutos). Los asteriscos en los demás campos significan "cualquier hora, día, mes y día de la semana".

Verifica rutas reales

Asegúrate de usar las rutas absolutas correctas. Para saber dónde está el binario de PHP, ejecuta `which php`. Para `wp-cron.php`, usa `pwd` desde el directorio de WordPress.

Evita ejecutar como root

Por seguridad, no uses el crontab de root a menos que sea estrictamente necesario. Ejecuta el cron con el mismo usuario que el servidor web para respetar permisos de archivos.

Comprueba permisos

El archivo `wp-cron.php` debe ser accesible y ejecutable por el usuario del cron. Verifica con `ls -l wp-cron.php` y ajusta con `chmod` si hace falta.

Guarda y verifica

Guarda el archivo (en nano, pulsa Ctrl+O, Enter, Ctrl+X). El sistema confirmará que se instaló una nueva tarea cron. Para listar todas las tareas activas del usuario actual, ejecuta:

```
crontab -l
```

Deberías ver tu nueva línea listada. A partir de este momento, el servidor ejecutará `wp-cron.php` cada 10 minutos sin depender de visitas.

Paso 3: Verifica que el cron funciona correctamente

Ahora que has desactivado WP-Cron y creado el cron real, es crucial **verificar que todo funciona** antes de dar por terminada la configuración. Hay tres métodos complementarios que te recomiendo usar para confirmar que las tareas se ejecutan según lo previsto.

1. Observa el registro de tareas

Instala un plugin como **WP Cronrol** o **Advanced Cron Manager** desde el repositorio oficial de WordPress. Estos plugins te muestran todas las tareas cron programadas, cuándo se ejecutaron por última vez y cuándo está prevista la siguiente ejecución.

Accede a Herramientas → Cron Events (o similar) y comprueba que:

- Las tareas tienen una fecha de "última ejecución" reciente
- No hay tareas acumuladas con errores
- Los intervalos de ejecución coinciden con lo configurado



2. Ejecuta wp-cron.php manualmente

Para forzar una ejecución inmediata y ver si hay errores, abre una terminal (o usa el navegador) y ejecuta:

```
curl -s https://[[tu-dominio.com]]/wp-cron.php?doing_wp_cron
```

Si todo está bien configurado, el comando terminará sin mostrar errores. Si recibes un código HTTP 403, 404 o 500, hay un problema de configuración (firewall, permisos, etc.) que debes resolver antes de continuar.

3. Comprueba funcionalidades críticas

• Publicaciones programadas

Programa un post para dentro de 5 minutos y verifica que se publica automáticamente sin necesidad de visitas.

• Emails automáticos

Envía un formulario de contacto o newsletter de prueba y confirma que los emails salen a tiempo.

• Backups automáticos

Revisa el plugin de backup para asegurarte de que la última copia se creó según el horario configurado.

Si todas estas comprobaciones pasan sin problemas, tu cron real está funcionando correctamente. Puedes pasar a las opciones avanzadas para refinar la seguridad y el rendimiento, o directamente al checklist imprimible para documentar tu configuración.

Opciones avanzadas: seguridad y rendimiento

Una vez que tienes el cron real funcionando, hay varias mejoras que puedes implementar para aumentar la seguridad y optimizar el rendimiento de tu instalación WordPress. Estas técnicas son opcionales pero altamente recomendadas para entornos de producción.

Bloquear acceso público a wp-cron.php

Si usas el método HTTP (cURL) para invocar el cron, cualquiera puede acceder a `wp-cron.php` desde un navegador y forzar la ejecución de tareas. Para evitar abusos, puedes restringir el acceso solo a la IP del servidor.

Añade estas líneas al archivo `.htaccess` de WordPress (justo después de `# BEGIN WordPress`):

```
Order Deny,Allow
Deny from all
Allow from 127.0.0.1
Allow from [[IP-servidor]]
```

Sustituye `[[IP-servidor]]` por la IP pública de tu servidor (la encontrarás en el panel de hosting o con `curl ifconfig.me` desde SSH). Esto bloquea todas las peticiones externas pero permite que el propio servidor ejecute el cron.

Evitar simultaneidad con archivos de bloqueo

Si tu hosting ejecuta el cron mientras todavía está en marcha una ejecución anterior (porque hay tareas pesadas), pueden producirse conflictos. Usa un archivo de bloqueo (lock file) para evitar ejecuciones simultáneas:

```
* /10 * * * * flock -n /tmp/wp-cron.lock /usr/bin/php -q /var/www/[[tu-sitio]]/public_html/wp-cron.php > /dev/null 2>&1
```

El comando `flock -n` crea un archivo temporal y solo ejecuta el cron si ese archivo no existe (es decir, si no hay otra ejecución en curso). Cuando termina, el archivo se elimina automáticamente.



WP-CLI para mayor control

Si tienes WP-CLI instalado, puedes usar un comando más específico que solo ejecuta los eventos pendientes, sin invocar toda la lógica de WordPress. Esto reduce la carga y mejora el tiempo de respuesta.



Registrar ejecuciones en un log

Cambia la redirección final para guardar un historial de ejecuciones. Útil para depuración y auditorías. El archivo crecerá con el tiempo, así que rótalos periódicamente.

```
* /10 * * * * wp cron event run --due-now --path=/var/www/[[tu-sitio]]/public_html >> /home/[[usuario]]/cron.log 2>&1
```

Esta línea ejecuta solo los eventos que están pendientes (`--due-now`), guarda toda la salida en `cron.log` y permite diagnosticar problemas si algo falla. Revisa el log cada semana y limpia las entradas antiguas para no ocupar espacio innecesario.



Consejo profesional: En sitios con muchísimo tráfico o tareas muy pesadas (procesamiento de imágenes, sincronización con APIs externas), considera aumentar el intervalo del cron a 15 o 30 minutos, o incluso dividir las tareas en múltiples cron jobs con horarios escalonados.

Checklist imprimible y troubleshooting rápido

Checklist de verificación (marca cada punto)

1

DISABLE_WP_CRON añadido
La constante está en wp-config.php antes del comentario "stop editing"

2

Cron creado (/5-15 min)
Panel de hosting o crontab configurado con intervalo entre 5 y 15 minutos

3

Ruta/URL verificada
Comando usa rutas absolutas correctas o URL completa con ?doing_wp_cron

4

Tareas pendientes vacías
WP Cronrol muestra que no hay eventos acumulados con errores

5

Publicaciones programadas OK
Post de prueba programado se publicó automáticamente en el horario previsto

6

Emails de formularios OK
Los emails de contacto, notificaciones y newsletters se envían puntualmente

7

Backup reciente guardado
Tienes una copia de seguridad completa antes de hacer los cambios

Troubleshooting: soluciones a problemas comunes

Síntoma	Causa probable	Solución
El cron no se ejecuta	Ruta PHP incorrecta o permisos denegados	Verifica con which php y ajusta permisos con chmod +x
Doble ejecución de tareas	WP-Cron no está deshabilitado o hay cron duplicado	Confirma DISABLE_WP_CRON en wp-config y elimina crons duplicados
HTTP 403 o 404 al invocar wp-cron.php	Firewall, WAF o reglas .htaccess bloquean la petición	Añade excepción en firewall/WAF o ajusta .htaccess para permitir IP del servidor
Tiempo de ejecución demasiado largo	Tareas pesadas que exceden límites de tiempo del servidor	Aumenta intervalo a 15-30 min o divide tareas en múltiples cron jobs escalonados
Emails no salen aunque el cron corra	Problema de configuración SMTP, no del cron	Revisa plugin SMTP (WP Mail SMTP) y logs de email del servidor
El log cron.log crece sin control	No hay rotación de logs configurada	Usa logrotate o limpia manualmente cada mes con : > cron.log



Recuerda: Si algo falla y no sabes cómo solucionarlo, puedes revertir todos los cambios eliminando la línea `DISABLE_WP_CRON` de wp-config.php y borrando la tarea cron del servidor. WordPress volverá al sistema basado en visitas y todo funcionará como antes (con sus limitaciones, pero de forma predecible).

Hoja de control y cierre: reloj real, tareas a tiempo

Para mantener tu cron en óptimas condiciones, te recomiendo revisar periódicamente (una vez al mes) que todo sigue funcionando correctamente. Usa esta tabla para llevar un registro de verificaciones:

Hoja de control (30 días)

Fecha	Intervalo	Evento crítico	¿Ejecutó?	Errores	Notas
01/01	10 min	Backup automático	✓	—	OK
08/01	10 min	Newsletter semanal	✓	—	Enviado 9:00
15/01	10 min	Publicación programada	✓	—	Post en vivo
22/01	10 min	Sincronización API	✗	Timeout	Aumentar límite
29/01	15 min	Backup + limpieza	✓	—	Intervalo ajustado

Si detectas algún error, consulta la sección de troubleshooting y ajusta la configuración según sea necesario. La mayoría de problemas se solucionan aumentando el intervalo o mejorando los permisos de archivos.

Conclusión: autonomía y fiabilidad

Reloj real = tareas a tiempo.

Has pasado de un sistema dependiente de visitas a un cron profesional que garantiza la ejecución puntual de todas las tareas programadas. Tu WordPress ahora funciona de forma autónoma, sin importar el tráfico.

Más guías en [fabini.one](#)

[@fabinidc](#)

Nota legal: Esta guía se proporciona "tal cual", sin garantías de ningún tipo. El autor no se hace responsable de daños o pérdidas derivadas de la aplicación de estas instrucciones. Siempre mantén backups actualizados y prueba en un entorno de desarrollo antes de aplicar cambios en producción.

Si esta guía te ha resultado útil, compártela y etiqueta [@fabinidc](#) en redes sociales. Sigamos mejorando WordPress juntos.