

Alerta mundial por dos graves fallas en PHP: ¿por qué están desconectando sitios web?

El equipo de desarrollo de PHP publicó el 21 de octubre cuatro actualizaciones de emergencia para corregir dos vulnerabilidades críticas que permiten ejecutar código de forma remota sin necesidad de autenticación. Identificadas como CVE-2026-109618 y CVE-2026-109619, ambas recibieron una calificación de 9,2 sobre 10 en CVSS v4.0 y son conocidas en conjunto como PHP2Shell.

El problema afecta al propio intérprete de PHP, por lo que no depende necesariamente de que una página o aplicación tenga errores en su programación. **Las vulnerabilidades están presentes desde PHP 5.4** y, según sus desarrolladores, modificar la configuración del servidor puede reducir ciertos riesgos, pero no ofrece una protección completa sin instalar las correcciones.



¿Qué es PHP y por qué su seguridad afecta a tantos sitios web?

PHP es un lenguaje de programación de código abierto y propósito general, utilizado principalmente para desarrollar sitios web y aplicaciones que necesitan procesar información en el servidor. A diferencia de JavaScript cuando se ejecuta en el navegador, PHP procesa las instrucciones antes de enviar la página al visitante, quien recibe el HTML generado sin tener acceso al código que realizó ese trabajo.

Esta tecnología se encuentra detrás de plataformas ampliamente utilizadas, como **WordPress, Drupal, Joomla, MediaWiki y Magento**, entre muchas otras. Por esa razón, una vulnerabilidad en el intérprete puede comprometer aplicaciones que funcionan correctamente y que no contienen errores propios relacionados con el ataque.

Su utilización va bastante más allá de mostrar páginas web, porque permite procesar formularios, administrar sesiones y cookies, consultar bases de datos y generar contenido dinámico a partir de la información que recibe el servidor. Para trabajar con motores como MySQL dispone de extensiones específicas y de PDO, una interfaz que permite acceder a distintos sistemas de bases de datos mediante una estructura común.

PHP también puede ejecutar programas desde la línea de comandos, y es compatible con Linux, Unix, Windows y macOS. Además, puede trabajar con servidores como Apache y Nginx mediante diferentes modalidades de ejecución, entre ellas módulos específicos y procesadores CGI.

Presencia de PHP en Internet y versiones utilizadas

Su presencia en Internet permite dimensionar el alcance de

PHP2Shell. De acuerdo con W3Techs, se utilizaba en el **69,3 % de los sitios web** cuyo lenguaje de programación del servidor podía identificarse.

Según W3Techs, el 65,3 % de los sitios que utilizan PHP opera con la rama 8, mientras que el 26,9 % permanece en PHP 7 y el 7,7 % utiliza PHP 5. Esto significa que **más de un tercio funciona con ramas que ya no reciben mantenimiento oficial**, a lo que se suman las instalaciones de PHP 8 que también utilizan versiones fuera de soporte.

¿Cómo funcionan las dos vulnerabilidades de PHP2Shell?

Ambas fallas están relacionadas con el manejo de la memoria y pertenecen a la familia conocida como **use-after-free**, que ocurre cuando un programa intenta utilizar información almacenada en una zona de memoria que ya fue liberada. Aunque comparten esa característica, cada vulnerabilidad llega al problema mediante una secuencia diferente de operaciones internas.

La situación es especialmente delicada cuando PHP funciona como módulo de Apache, modalidad conocida como mod_php. El equipo de seguridad confirmó que, bajo esa configuración, un atacante puede activar una de las rutas de explotación **antes de que el servidor ejecute siquiera la primera línea de código de la aplicación web**.

CVE-2026-109618: corrupción de memoria durante una petición

La primera vulnerabilidad aparece cuando PHP agota la memoria disponible mientras procesa una petición y algunos de sus datos internos todavía no han terminado de inicializarse. Si una operación de reserva de memoria falla en ese momento, el intérprete interrumpe el procesamiento mediante un mecanismo

que no registra adecuadamente el estado incompleto de esos datos.

El problema continúa durante el cierre de la petición, cuando PHP intenta liberar los recursos utilizados y encuentra estructuras de información que no quedaron correctamente preparadas. El procedimiento puede interpretar esos datos inválidos como si fueran correctos y utilizarlos en operaciones posteriores, una condición que los investigadores consiguieron transformar en ejecución remota de código.

CVE-2026-109619: información que permanece entre peticiones

La segunda vulnerabilidad tiene su origen en ciertos estados globales que PHP conserva mientras un proceso trabajador, denominado worker, atiende sucesivas peticiones web. Esos estados pueden contener información sobre errores, sesiones u objetos que utilizan memoria perteneciente a una petición determinada, por lo que deben limpiarse antes de liberar esa memoria.

Cuando ocurre un error fatal, PHP puede interrumpir una operación antes de completar esa limpieza y dejar referencias que apuntan a información que ya no existe. Si el mismo worker recibe otra petición, determinadas operaciones pueden intentar utilizar esas referencias antes de que el proceso de inicialización las sustituya, provocando el acceso a memoria liberada.

El aviso de seguridad de PHP identifica **dos mecanismos de explotación confirmados**. Uno utiliza las trazas de errores fatales (`fatal_error_backtraces`) presentes en PHP 8.5, mientras que el otro aprovecha el seguimiento del progreso de subida de archivos asociado a las sesiones (`session.upload_progress`), una funcionalidad que está activada por defecto.

Las correcciones para ambas vulnerabilidades quedaron incorporadas en **PHP 8.5.12, 8.4.27, 8.3.36 y 8.2.35**, publicadas el 21 de octubre. El equipo de seguridad advierte que no existe una combinación de ajustes de configuración capaz de eliminar todas las rutas de explotación, por lo que considera necesaria la actualización del intérprete.

¿Cómo se descubrió PHP2Shell y por qué se adelantó su divulgación?

La investigación comenzó dentro de **Patch the Planet**, una iniciativa de OpenAI a la que se incorporó Calif para examinar posibles vulnerabilidades en proyectos de software ampliamente utilizados. Wiz Research también identificó las vulnerabilidades de manera independiente durante agosto y las reportó al equipo de seguridad de PHP el 24 de ese mes.

Calif encontró las primeras rutas de ejecución remota de código el 1 de agosto y comunicó formalmente la primera vulnerabilidad el 11 de ese mes. El segundo reporte llegó el 21 de agosto y los participantes acordaron mantener una **divulgación coordinada hasta el 22 de octubre**, con el propósito de permitir que los desarrolladores prepararan las correcciones.

Ryan Austin, investigador de Calif y autor del análisis técnico de PHP2Shell, explicó que el conocimiento previo de las vulnerabilidades del lenguaje fue determinante para orientar la investigación. También aclaró que el descubrimiento inicial no necesitó herramientas exclusivas ni modelos de lenguaje que estuvieran fuera del alcance del público.

«No se usó ningún harness especial ni acceso a modelos que no estuvieran disponibles para todo el mundo en el descubrimiento inicial del bug. [...] Ningún modelo es simplemente capaz de desenterrar ‘todos los bugs’ con prompts

sencillos. Al menos por ahora».

Ryan Austin, investigador de Calif

La filtración del 20 de octubre y los primeros intentos de ataque

El calendario de divulgación cambió el **20 de octubre**, cuando comenzaron a circular detalles de las vulnerabilidades en redes sociales antes de la fecha acordada. Ese mismo día, el equipo de seguridad de PHP anunció una actualización extraordinaria para el **21 de octubre a las 14:00 UTC**, mientras que watchTowr fue la primera firma en confirmar públicamente la autenticidad de las fallas.

Los intentos de explotación aparecieron **incluso antes de que los parches estuvieran disponibles**, según los registros de las empresas que monitoreaban la actividad. Amazon detectó un primer intento el 17 de octubre en su red de honeypots y varios exploits de prueba desde cerca de las 02:00 UTC del 21, unas doce horas antes de que saliera la versión corregida.

La firma watchTowr detectó intentos de explotación el 21 de octubre mediante su red de honeypots, sistemas preparados para registrar ataques informáticos. Hasta el 22 de octubre, **ninguna de las dos vulnerabilidades había sido incorporada al catálogo** de vulnerabilidades explotadas conocidas de la Agencia de Seguridad de Infraestructura y Ciberseguridad de Estados Unidos (CISA).

En los Países Bajos, la preocupación por el alcance de PHP2Shell llevó a varias organizaciones a **desconectar temporalmente sus sitios web** mientras esperaban las actualizaciones. Entre ellas estuvieron el medio tecnológico Tweakers, la Cámara de Representantes, el Centro Nacional de Ciberseguridad (NCSC) y el portal especializado Security.NL.

¿Qué servidores y aplicaciones están expuestos a PHP2Shell?

El alcance de las vulnerabilidades no se limita a una configuración particular del intérprete, ya que el equipo de PHP considera expuestas las instalaciones que atienden peticiones web mediante **PHP-FPM, Apache con mod_php, php-cgi, FrankenPHP o LiteSpeed**. La gravedad puede variar según la modalidad de ejecución, las características del servidor y las condiciones necesarias para completar el ataque.

Durante sus pruebas, Calif consiguió ejecutar código de forma remota y sin autenticación en instalaciones de versiones vigentes de **WordPress, Drupal, Joomla, MediaWiki, Roundcube, Magento, phpMyAdmin, Nextcloud, Laravel y Symfony**. Los investigadores también demostraron la explotación contra scripts PHP arbitrarios en determinados escenarios, lo que confirma que el problema reside en el funcionamiento del propio lenguaje.

Esto no significa que todas esas aplicaciones tengan una vulnerabilidad independiente ni que cualquier servidor pueda comprometerse mediante una única petición idéntica. La explotación depende de las condiciones de cada instalación, aunque un sitio con su código correctamente desarrollado puede seguir expuesto si utiliza una versión vulnerable del intérprete.

Las versiones antiguas de PHP complican la actualización

El principal problema de mantenimiento aparece en las instalaciones que utilizan ramas de PHP fuera de soporte, porque las actualizaciones oficiales solo están disponibles para las versiones 8.2 y posteriores. **PHP 8.1 y todas las ramas anteriores ya no reciben correcciones regulares** del proyecto, aunque algunas distribuciones Linux y proveedores de

soporte extendido pueden ofrecer parches adaptados a sus sistemas.

Debido a la gravedad de PHP2Shell, el equipo de seguridad también publicó adaptaciones mínimas del código corregido para **PHP 7.4, 8.0 y 8.1**. Estas versiones están dirigidas principalmente a administradores que compilan PHP por su cuenta o que no disponen de un proveedor de soporte extendido capaz de entregar los arreglos.

Para **PHP 7.3 y versiones anteriores, incluidas las instalaciones de PHP 5**, el proyecto no publicó correcciones equivalentes. En esos sistemas, la alternativa indicada es migrar a una versión que continúe recibiendo actualizaciones de seguridad.

Incluso PHP 8.2 tiene un plazo de mantenimiento limitado, porque alcanzará el final de su vida útil el **31 de diciembre de 2026** y dejará de recibir parches oficiales después de esa fecha. Por lo tanto, aunque PHP 8.2.35 corrige ambas vulnerabilidades, las instalaciones que permanecen en esa rama también deben considerar una actualización posterior.

Distribución de parches y actualización de PHP

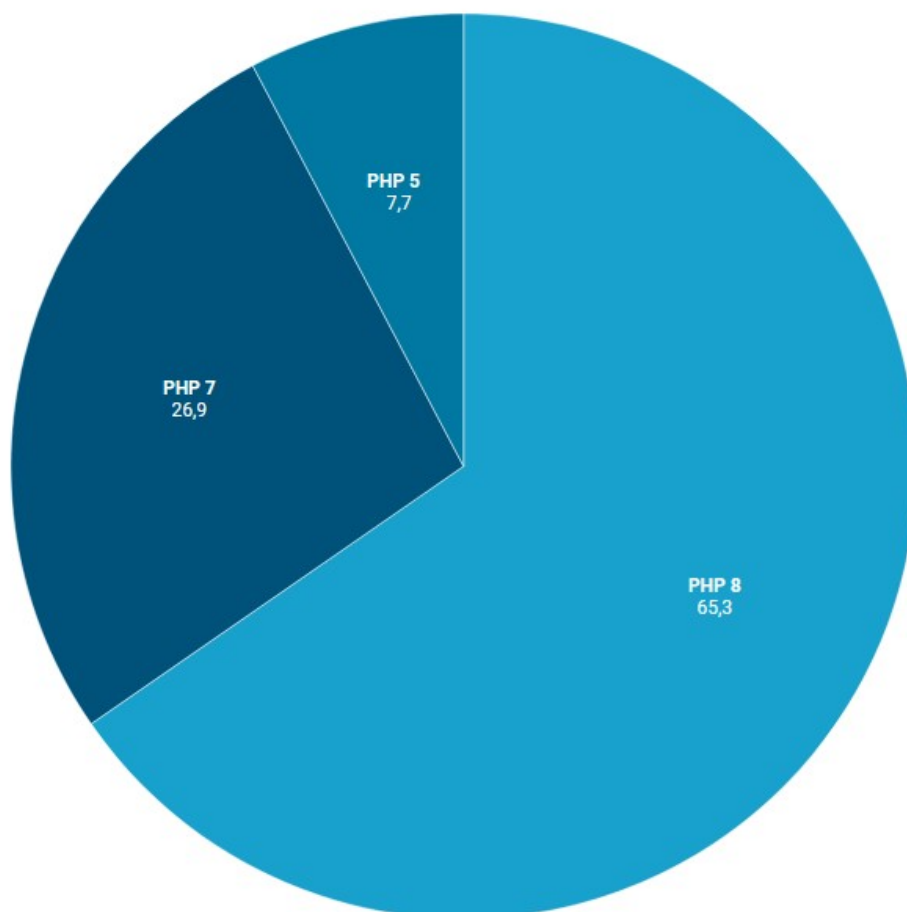
La distribución de los paquetes agregó otra dificultad durante el lanzamiento, especialmente para quienes utilizan el repositorio comunitario `rpms.remirepo.net`, habitual en sistemas basados en Red Hat. El equipo de PHP informó que ese repositorio **no tendría las actualizaciones disponibles durante la jornada del 21 de octubre** y recomendó utilizar los paquetes de la distribución Linux correspondiente o compilar directamente desde el código fuente.

Mientras se realiza la actualización, existen ajustes que reducen la exposición a determinados procedimientos de explotación, pero **ninguno corrige completamente las dos**

vulnerabilidades. Las versiones publicadas el 21 de octubre contienen los cambios necesarios en el intérprete, por lo que instalar una de las versiones corregidas sigue siendo la medida indicada por el equipo de seguridad de PHP.

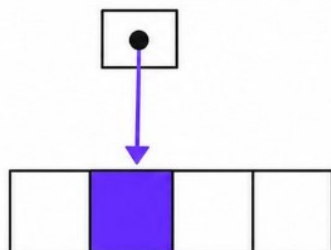
Versiones de PHP

■ PHP 8 ■ PHP 7 ■ PHP 5

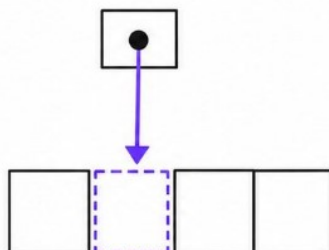


¿Qué es un use-after-free?

1. PHP reserva memoria

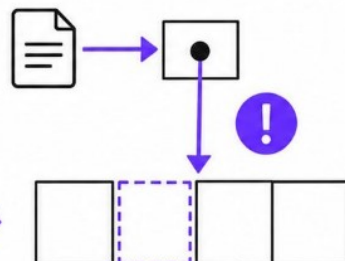


2. PHP libera ese espacio



Memoria liberada

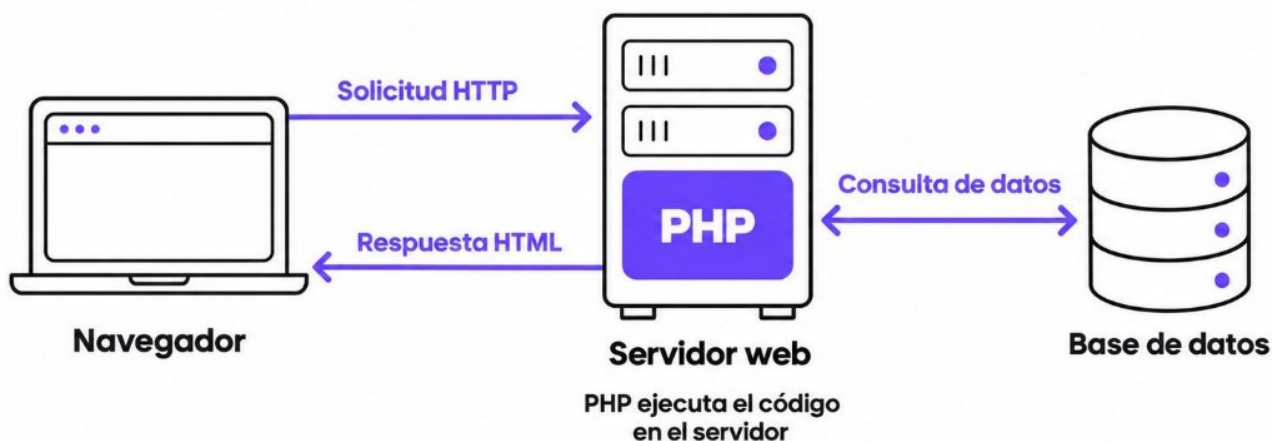
3. Otra petición usa la referencia



Acceso inválido

El programa intenta acceder a memoria que ya fue liberada

¿Cómo funciona PHP?



El navegador recibe HTML, no el código PHP.

PHP2Shell: del hallazgo al parche

Siete hitos entre el 1 de agosto y el 22 de octubre de 2026



Los intentos de explotación comenzaron antes de la publicación de los parches.

Las separaciones entre hitos son esquemáticas y no representan intervalos de tiempo proporcionales.

¿Qué versión de PHP tengo y qué hago?

Medidas frente a las vulnerabilidades CVE-2026-109618 y CVE-2026-109619

VERSIÓN INSTALADA	QUÉ HACER
PHP 8.2 a 8.5 Corrección oficial disponible	ACTUALIZAR Instalar la versión corregida de la misma rama 8.2.35 8.3.36 8.4.27 8.5.12
PHP 7.4, 8.0 y 8.1 Sin actualización oficial de rama	APLICAR CORRECCIÓN Usar el backport mínimo de código fuente o el parche del proveedor con soporte extendido
PHP 7.3 o anterior Incluye PHP 5	MIGRAR Pasarse a una versión con mantenimiento vigente No existe corrección para estas ramas

Las medidas de configuración pueden reducir la exposición, pero no sustituyen las actualizaciones de seguridad.

Fuente: pisapapeles.