

What is PHP2Shell?

PHP2Shell is the name for two remote code execution vulnerabilities in PHP core, CVE-2026-109618 (memory corruption when PHP runs out of memory mid-request) and CVE-2026-109619 (request state that survives into the next request on the same worker). Both are rated 9.2 on CVSS v4, need no authentication, and trigger during request parsing, before application code runs. Fixed in PHP 8.5.12, 8.4.27, 8.3.36 and 8.2.35 on October 21, 2026; exploitation attempts were seen before disclosure.

HOW IT WORKS

01 How the attack works

The entry point is an ordinary HTTP request. The project's own advisory describes a demonstrated exploit against Apache with mod_php where an unauthenticated request triggers the attack before any application code runs. Shown for defensive context, the shape a defender should recognize is:

- A ``multipart/form-data`` POST to any PHP endpoint, including ones that never expect uploads, built to push PHP to its `memory_limit` while the body is being parsed.
- Requests that carry a ``PHP_SESSION_UPLOAD_PROGRESS`` field, which drives the default-on upload-progress tracking involved in CVE-2026-109619.
- Repeated attempts against the same worker pool, since the stale-state path depends on a later request landing on the same PHP worker.

Every common deployment is in scope: PHP-FPM, Apache mod_php (rated highest risk), php-cgi, FrankenPHP and LiteSpeed. Amazon threat intelligence reported an exploitation attempt in its honeypots on October 17, 2026, four days before the fixes shipped, and broad exploitation attempts followed disclosure.

CLEAN APPLICATION CODE DOES NOT HELP

The bug fires while PHP parses the request, before your router, authentication or WAF-aware middleware run. A fully patched CMS on an unpatched PHP binary is still exposed.

02 How to check if you are exposed

Check the PHP binary that actually serves traffic, on every host and inside every container image:

HOW TO DEFEND

- Set ``session.upload_progress.enabled=0`` if you do not use upload progress. On PHP 8.5, also set `fatal_error_backtraces=0`.
- Set ``memory_limit=-1`` administratively (for example `php_admin_value`) so attackers cannot trigger the out-of-memory path, and cap memory with a container or cgroup limit instead. Do not use `RLIMIT_AS` or `RLIMIT_DATA`.
- Keep ``post_max_size`` conservative and enforce request size, part count and header limits at the reverse proxy.
- Block ``multipart/form-data`` at the edge on applications that never accept uploads.

SOURCES

- [1] PHP security advisory GHSA-w8hw-w9fx-j79w (CVE-2026-109618)
- [2] PHP security advisory GHSA-6wmf-4rv5-p3rp (CVE-2026-109619)
- [3] NVD: CVE-2026-109618
- [4] NVD: CVE-2026-109619

- Version: `php -v`, or for FPM `php-fpm -v`
• Vulnerable if below 8.2.35, 8.3.36, 8.4.27 or 8.5.12 on those branches. Branches 8.1 and older have no official upstream fix.
- Upload progress: `php -i | grep session.upload_progress.enabled`. A value of On means the CVE-2026-109619 path is open.
- Distribution packages: your OS vendor may ship a patched build with an older-looking version number. Check the vendor advisory for the package release that contains the fix.
- Logs to hunt: multipart POSTs followed by Allowed memory size exhausted fatal errors, bursts of HTTP 500s, or PHP-FPM workers exiting on SIGSEGV. Any web server process spawning `sh`, `curl` or `wget` is a strong compromise signal.

Find the path to a shell before an attacker does.

securelayer7.net/learn/web-exploitation/what-is-php2shell

[Open online](https://securelayer7.net/learn/web-exploitation/what-is-php2shell)