CERT-EU

Security Advisory 2026-018

Critical Vulnerabilities in PHP

2026-10-21 — v1.2

TLP:CLEAR

History:

- *20/10/2026 — v1.0 – Initial publication*
- *21/10/2026 — v1.1 – Updates with additional information*
- *23/10/2026 — v1.2 – Updates with information about a third vulnerabilty*

Summary

On 20 October 2026, the PHP project announced an upcoming security release addressing **two critical vulnerabilities** in PHP [1]. Public reports indicated that these vulnerabilities could lead to remote code execution (RCE) [2,3].

The **PHP team released the fixes today**, 21 October 2026 at 13:30 UTC, in PHP versions 8.5.12, 8.4.27, 8.3.36 and 8.2.35 [4]. Both vulnerabilities can allow **unauthenticated remote code execution**. Technical details are now being discussed publicly, and initial proofs of concept (PoCs) have begun to circulate. This raises the likelihood that attackers are already exploiting the vulnerabilities.

[New] On 23 October, PHP disclosed a new RCE vulnerability. However, it affects only web applications configured in an unusual way, and PHP considers it an unlikely risk. PHP will not issue an out-of-band release for this vulnerability [8]. Instead, organisations are advised to apply mitigation to fully protect from this RCE vector [4].

CERT-EU advises organisations to **patch affected assets immediately**. Where that is not feasible, organisations should consider isolating internet-facing applications from public networks until a fix is applied and verified.

Technical Details

The vulnerability **CVE-2026-109618** (GHSA-w8hw-w9fx-j79w), with a CVSS score of 9.2, allows remote code execution through memory corruption during out-of-memory (OOM) handling [4,6]. PHP enforces its user-configurable `memory_limit` by using `longjmp()` to abort any request that exceeds the limit, which raises an OOM error. This can interrupt PHP while internal request data is only partially initialised, for example, while parsing a form-based file upload. Request cleanup may then treat the invalid data as valid and try to free memory that was never initialised. That memory may still contain data from earlier allocations controlled by the attacker.

The vulnerability **CVE-2026-109619** (GHSA-6wmf-4rv5-p3rp), with a CVSS score of 9.2, allows remote code execution through stale global state [4,7]. Internal state owned by a request can

survive after its backing memory has been freed, for example after a fatal error. A later request handled by the same worker may use that stale state before normal initialisation resets it.

Every PHP installation that serves web requests is at risk, including PHP-FPM, Apache with `mod_php`, `php-cgi`, FrankenPHP and LiteSpeed. The PHP team considers `mod_php` the highest-risk setup. A demonstrated exploit shows that one unauthenticated request can trigger the attack before any application code runs [4].

[New] There is no detail available about the vulnerability **CVE-2026-115560** which could lead to remote code execution. The PHP team will publish a full advisory with the next release.

Affected Products

- PHP 8.5 versions prior to 8.5.12
- PHP 8.4 versions prior to 8.4.27
- PHP 8.3 versions prior to 8.3.36
- PHP 8.2 versions prior to 8.2.35
- End-of-life branches (PHP 8.1 and older), which no longer receive updates from the PHP project [4]

Many commercial and open-source products use PHP as a backend technology - these could also be affected, even if they do not directly expose a PHP installation. CERT-EU recommends consulting the NCSC-NL GitHub repository to identify third-party software that embeds PHP and may be impacted by these vulnerabilities [5].

Recommendations

CERT-EU strongly recommends updating all affected PHP installations **as soon as possible** to PHP 8.5.12, 8.4.27, 8.3.36 or 8.2.35 [4].

- **Distribution packages:** Linux distributions and LTS vendors received the patches in advance. Check with your vendor for updated packages.
- **End-of-life versions (PHP 8.1 and older):** Some distributions and LTS vendors provide backported fixes, so contact your vendor or upgrade to a supported version. Because of the severity, the PHP security team has also published minimal source backports for PHP 7.4, 8.0 and 8.1 for organisations that build from source or have no LTS vendor.
- **rpms.remirepo.net users:** This volunteer-maintained repository will not have updates on release day. The PHP team urges users to switch to distribution packages or build from source.

Where immediate patching is not feasible, organisations should consider isolating internet-facing resources from public networks until the fix is applied and verified.

Temporary Mitigation

Only an update fully fixes these vulnerabilities. The measures below reduce exposure to specific attack paths but **do not make an installation safe** [4].

CVE-2026-109618 (OOM corruption):

- Set `memory_limit=-1` in a way applications cannot override, such as `php_admin_value` in the FPM pool or Apache configuration. Check that no pool, directory or application setting puts a finite limit back. This removes the controllable memory limit failure, but an underlying allocation failure can still reach the vulnerable code path.

- Set a low `post_max_size`: `8M` for applications that do not need file uploads, `32M` if uploads are required. Keep `upload_max_filesize` below `post_max_size`.
- Consider enforcing limits at the web server or proxy level for decoded request size, input field count, multipart part count, header count and nesting depth. These limits must also cover requests without a usable `Content-Length`.
- Use a container or cgroup memory limit as an availability boundary, preferably with a policy that kills only the affected worker. Do not use `RLIMIT_AS` or `RLIMIT_DATA` instead, because they can trigger the vulnerable allocation failure.

CVE-2026-109619 (stale global state):

- Set `session.upload_progress.enabled=0` and restart every worker. Apply this in `php.ini`, an FPM pool or a server configuration that takes effect before request parsing. Do not rely on `ini_set()` or FPM `.user.ini`. This blocks the confirmed session paths but not other variants.
- On PHP 8.5, set `fatal_error_backtraces=0` and restart every worker. This blocks the confirmed fatal backtrace path only.
- Remove invalid or deprecated per-request settings that emit diagnostics before initialisation. Hiding diagnostics with `error_reporting`, `display_errors` or `@` does not help.
- Fix fatal errors and execution timeouts that attackers can reach, including failures in callbacks, upload or import processing, and cleanup.

[New] CVE-2026-115560:

- Set `session.upload_progress.enabled=0` and restart every worker until the next regular bugfix release, to fully protect from CVE-2026-115560.

References

[1] https://www.php.net/archive/2026.php#2026-10-20-1

[2] https://cyberplace.social/@GossiTheDog/117474220875882526

[3] https://x.com/watchtowrcyber/status/2112611507749982605

[4] https://www.php.net/index.php#2026-10-20-1

[5] https://github.com/NCSC-NL/php2shell

[6] https://github.com/php/php-src/security/advisories/GHSA-w8hw-w9fx-j79w

[7] https://github.com/php/php-src/security/advisories/GHSA-6wmf-4rv5-p3rp

[8] https://thehackernews.com/2026/10/urgent-php-warns-of-another-unpatched.html