

PHP2Shell Security Guidance

CampusGuard/RedLens Infosec | October 23, 2026

Action required: Update affected PHP installations immediately for CVE-2026-109618 and CVE-2026-109619. Review the temporary configuration guidance for CVE-2026-115560 until PHP publishes the next regular bugfix release. Exploitation attempts against CVE-2026-109618 and CVE-2026-109619 have been reported since October 21, 2026, and working exploit code exists for at least some PHP 8.4 releases. No exploitation of CVE-2026-115560 has been reported.

Patched and affected versions

PHP released fixes for CVE-2026-109618 and CVE-2026-109619 on October 21, 2026. CVE-2026-115560 was disclosed on October 23, 2026; PHP has not yet published affected version ranges or a patched release.

Affected version	Patched version
PHP 8.5 before 8.5.12	PHP 8.5.12
PHP 8.4 before 8.4.27	PHP 8.4.27
PHP 8.3 before 8.3.36	PHP 8.3.36
PHP 8.2 before 8.2.35	PHP 8.2.35

Older versions: PHP 8.1 and earlier are no longer supported by the PHP project. Use a distribution or LTS vendor package containing the backported fixes, apply PHP's source backport for PHP 7.4, 8.0, or 8.1 where appropriate, or move to a supported PHP release.

CVE-2026-109618

Remote code execution through out-of-memory corruption

- **Disclosed:** October 21, 2026
- **Affected versions:** PHP 8.2 before 8.2.35; PHP 8.3 before 8.3.36; PHP 8.4 before 8.4.27; PHP 8.5 before 8.5.12
- **Patched versions:** PHP 8.2.35, 8.3.36, 8.4.27, and 8.5.12
- **Additional guidance:**
 - PHP may leave request memory in an invalid state when an allocation fails during request handling.
 - For systems that cannot be patched immediately, PHP recommends:
 - Set `memory_limit=-1` administratively (for example, `php_admin_value` in the PHP-FPM pool or Apache configuration), confirm that no pool, directory, or application setting restores a finite limit, and restart all PHP workers.
 - Because this removes PHP's own memory cap, enforce memory limits at the container or cgroup level instead, ideally so that only the affected worker is terminated. Do not use `RLIMIT_AS` or `RLIMIT_DATA`, which can trigger the allocation failure this vulnerability depends on.
 - Keep `post_max_size` conservative, enforce request size and multipart limits at the web server or proxy, and block multipart/form-data at the frontend if the application does not accept it.
 - These measures reduce exposure but do not fully mitigate the vulnerability without patching.

CVE-2026-109619

Remote code execution through stale global state

- **Disclosed:** October 21, 2026
- **Affected versions:** PHP 8.2 before 8.2.35; PHP 8.3 before 8.3.36; PHP 8.4 before 8.4.27; PHP 8.5 before 8.5.12
- **Patched versions:** PHP 8.2.35, 8.3.36, 8.4.27, and 8.5.12
- **Additional guidance:**
 - Request-owned state may remain available to a later request handled by the same PHP worker.
 - Set `session.upload_progress.enabled = 0` in `php.ini`, the PHP-FPM pool configuration, or the web server configuration, then restart all PHP workers. Setting it with `ini_set()` or an FPM `.user.ini` file has no effect. On PHP 8.5, also set `fatal_error_backtraces = 0` the same way. File uploads continue to work; only upload progress tracking is disabled.
 - These measures address confirmed paths but do not provide complete protection without patching.

CVE-2026-115560

Remote code execution through session upload progress

- **Disclosed:** October 23, 2026
- **Affected versions:** Not yet specified by the PHP Security Team
- **Patched version:** Not yet released; planned for the next regular bugfix release
- **Additional guidance:**
 - PHP has not yet published affected version ranges or a patched version.
 - The issue requires an unusual configuration: `session.serialize_handler` is set to `php_binary` or `php_serialize`, the application changes it to `php` at runtime with `ini_set()` before calling `session_start()`, and session upload progress remains enabled.
 - Set `session.upload_progress.enabled = 0` as described under CVE-2026-109619. Systems that have already applied it require no additional action.
 - Applications that never change the session serialize handler at runtime can keep session upload progress enabled.

Recommended actions

1. **Inventory:** Identify PHP runtimes in web servers, containers, appliances, managed platforms, and embedded products.
2. **Prioritize:** Address Internet-facing services and systems accepting untrusted requests or file uploads first.
3. **Patch:** Install an upstream fixed release or a vendor package that explicitly includes fixes for both CVEs.
4. **Restart:** Restart PHP workers, PHP-FPM, Apache, containers, or the applicable service so the patched code is loaded.
5. **Verify:** Confirm the version of PHP actually serving requests, not just the command line. `php -v` reports the CLI binary, which can differ from PHP-FPM or the Apache module, so check `php-fpm -v` or a temporary `phpinfo()` page (delete it afterward). Distribution packages often backport fixes without changing the PHP version number, so check the package changelog or vendor advisory for both CVE numbers. Then test critical application functions.
6. **Monitor:** Neither the PHP project nor major vendors have published official indicators of compromise. Until they do, hunt in web, WAF, process, endpoint, and network telemetry for unusual or oversized multipart requests, unexpected child processes of PHP workers, new or modified files in web roots, and unexpected outbound connections.

Mitigating controls

Use these controls only while the update is being tested or deployed: `

- Apply the configuration mitigations listed under each CVE above.
- Restrict PHP applications that do not need public access to trusted networks or VPN.
- Use AWS WAF, Azure WAF, Cloudflare, or equivalent rules to block malformed or excessive multipart requests and known PHP2Shell patterns.
- Restrict Internet access to PHP administration interfaces and unnecessary outbound access from web workers.
- If you take a site offline, use a maintenance page or network restriction. Do not simply disable the PHP handler, which can cause the web server to serve PHP source code and credentials as plain text.
- Test each control before enforcing it to avoid interrupting required application functions.

Industry guidance

- [PHP Informational Advisory: CVE-2026-115560 session upload progress](#)
- [PHP Security Advisory: Critical remote code execution vulnerabilities](#)
- [PHP Advisory: CVE-2026-109618, RCE via OOM corruption](#)
- [PHP Advisory: CVE-2026-109619, RCE via stale global state](#)
- [AWS Security Blog: PHP2Shell WAF mitigations](#)
- [Microsoft Azure Network Security: PHP2Shell upgrade and WAF guidance](#)
- [NCSC-NL: PHP2Shell operational guidance](#)

CampusGuard/RedLens Infosec position: Patch affected systems as soon as possible. Interim controls may reduce exposure, but they do not replace the update.