

THREAT ADVISORY

PHP2Shell: Critical PHP Vuln Expose Web Servers to RCE

Updated October 26, 2026

Threat Level	CRITICAL
CVE Identifiers	CVE-2026-109618; CVE-2026-109619
CVSS v4.0	9.2 / 10 (Critical)
Attack Type	Pre-authentication / unauthenticated Remote Code Execution (RCE)
Affected Technology	PHP installations serving web requests, including PHP-FPM, Apache mod_php, php-cgi, FrankenPHP and LiteSpeed
Immediate Action	Inventory PHP exposure, patch or apply vendor backports, and hunt for exploitation attempts

Executive Summary

Two critical vulnerabilities in the PHP interpreter, CVE-2026-109618 and CVE-2026-109619, collectively referred to as PHP2Shell, can allow an unauthenticated attacker to achieve remote code execution in vulnerable web-serving PHP environments. The PHP Project released emergency fixes on October 21, 2026. Public reporting indicates exploitation attempts were observed before the patches were released, including an attempt reported by Amazon Threat Intelligence on October 17, 2026.

The risk extends beyond conventional websites. PHP is embedded in content-management systems, internet-facing business applications, containers, hosting platforms, appliances, network-device management interfaces, and vendor products. Apache mod_php deployments are considered the highest-risk PHP configuration, although PHP-FPM, php-cgi, FrankenPHP and LiteSpeed are also affected.

Blackswan's assessment: Treat PHP2Shell as an emergency vulnerability-management and threat-hunting priority. Patching is necessary, but an organization that finds an exposed or previously vulnerable system should also review telemetry for signs of exploitation. A patched host that was already compromised cannot be assumed clean solely because the PHP runtime was updated.

What Changed Since the Initial Advisory

- **Confirmed exploitation attempts:** security reporting now documents exploitation activity, including attempts observed before public patch release.
- **Affected execution modes are better defined:** all PHP installations serving web requests are at risk, with Apache mod_php identified as the highest-risk deployment model.
- **End-of-life PHP requires special handling:** PHP 8.1 and earlier do not receive normal supported binary releases from the PHP Project. Source backports exist for PHP 7.4, 8.0 and 8.1, while operating-system and LTS vendors may provide patched packages.
- **The exposure is broader than websites:** embedded PHP in appliances, network devices, management interfaces and third-party products must be assessed separately.
- **Public scanning is evolving:** exposure tools can help with triage, but version-only tests can produce misleading results when vendors have backported fixes without changing the displayed PHP version.
- **Important nuance:** public research differs on exploitability at scale. Some researchers describe broad default-configuration impact, while Sansec reports that public proof-of-concept exploitation often requires target-specific conditions. This does not change the recommendation to patch immediately.

Vulnerability Overview

CVE	CVSS v4.0	Technical Issue	Potential Impact
CVE-2026-109618	9.2	Out-of-memory handling can leave request memory in an uninitialized state during cleanup.	Remote code execution
CVE-2026-109619	9.2	Request-owned global state can reference freed memory and be reused by a later request.	Remote code execution

Both vulnerabilities occur in the PHP engine itself. Application code does not have to contain the underlying flaw for a vulnerable PHP runtime to create exposure.

Affected Versions and Fixed Releases

PHP Branch	Fixed Release	Blackswan Guidance
8.5	8.5.12	Update immediately.
8.4	8.4.27	Update immediately.
8.3	8.3.36	Update immediately.
8.2	8.2.35	Update immediately.

PHP 8.1 and earlier are end-of-life under normal PHP Project support. Organizations running older branches should validate operating-system, hosting-provider, appliance-vendor or LTS backports and plan migration to a supported PHP branch. Do not rely on the web banner or displayed version alone when a vendor may have backported the security fix.

Technology and Application Exposure

Any PHP installation that serves web requests should be inventoried. Priority areas include:

- Apache with mod_php (highest-risk deployment model reported by PHP/security advisories)
- PHP-FPM and php-cgi
- FrankenPHP and LiteSpeed
- WordPress, Drupal, Joomla, Magento / Adobe Commerce, phpMyAdmin, Nextcloud and other PHP applications
- Custom PHP applications and internal administrative portals
- Containers and cloud images that package PHP
- Network appliances, management interfaces and third-party products that embed PHP

Threat Activity and Exploitation Status

Security reporting after the October 21 patch release indicates that exploitation attempts have been observed. Sophos cites Amazon Threat Intelligence reporting an exploitation attempt on October 17, 2026, before public disclosure, and reports that other vendors subsequently observed additional attempts. SecurityWeek likewise updated its reporting to note exploitation attempts, including activity before fixes landed.

At the same time, defenders should distinguish observed exploitation attempts from confirmed compromise of a particular environment. Palo Alto Networks Unit 42 retracted an earlier attribution of post-compromise activity to PHP2Shell after it could not confirm that initial access was associated with these CVEs. This reinforces the need to preserve evidence and validate the initial-access vector rather than assume attribution from vulnerable-PHP presence alone.

Immediate Defensive Actions

1. **Inventory affected assets.** Identify PHP runtimes from running processes, package inventories, containers, cloud workload images, application stacks, appliances and vendor products. Prioritize internet-facing systems.
2. **Patch supported PHP branches.** Upgrade to PHP 8.5.12, 8.4.27, 8.3.36 or 8.2.35, or apply the equivalent security package supplied by the operating-system, hosting or appliance vendor.
3. **Address end-of-life PHP.** Validate vendor/LTS backports where available and move unsupported deployments to a supported branch as quickly as operationally possible.
4. **Remove or restrict exposure when patching cannot be completed.** Consider taking a vulnerable internet-facing service offline or restricting access until an effective security update is installed.
5. **Validate the running runtime after remediation.** Confirm the active PHP binary/package and restart affected workers/services as required by the platform or vendor update procedure.
6. **Threat hunt.** Review web, application, EDR/XDR, WAF, reverse-proxy, network and cloud telemetry for exploit attempts and suspicious post-request activity.
7. **Engage vendors.** Require third-party hosting, SaaS, appliance and managed-service providers to confirm whether PHP is embedded, whether the product is affected, and the remediation status.

SOC Detection and Threat Hunting

Blackswan recommends correlating application-layer activity with endpoint and network telemetry. The following are investigation leads rather than standalone proof of PHP2Shell exploitation:

Web / Application Telemetry

- Unusual or malformed requests reaching PHP endpoints
- Unexpected HTTP 500 responses, PHP worker crashes, out-of-memory errors or abnormal application behavior
- Repeated requests against internet-facing PHP applications from unfamiliar infrastructure
- Requests followed closely by anomalous child-process or outbound-network activity

Endpoint / EDR / XDR Telemetry

- Unexpected shell or command-interpreter processes spawned by php, php-fpm, Apache/httpd, nginx-adjacent application workers or other web-service processes
- Unexpected executable or script creation in web roots, temporary directories or application directories
- New persistence mechanisms, unauthorized scheduled activity, suspicious service changes or credential-access behavior on the web server
- Signs of lateral movement originating from an internet-facing application server

Network / SIEM Correlation

- Unexpected outbound connections from PHP/web servers following suspicious inbound requests
- New internal connections from an internet-facing web server to identity, database, file or management systems
- Correlation of vulnerable-asset inventory with WAF/IDS/IPS detections and endpoint process telemetry
- High-priority review of exposed hosts that showed suspicious PHP or web-service activity before patching

Incident Response Considerations

If exploitation is suspected, activate established incident-response procedures. Preserve logs and endpoint telemetry before rebuilding or wiping affected systems when feasible. Scope the event beyond the vulnerable PHP process: successful RCE may provide an attacker the privileges of the web-server account and can become an initial-access point for persistence, credential theft, data access or lateral movement.

Do not use "patched" as the closure criterion for a suspected compromise. Validate that unauthorized code execution and persistence are absent, rotate credentials or secrets exposed to the affected application where warranted, and re-establish trust in the host before returning it to normal service.

MITRE ATT&CK Mapping

Technique	Relevance to Investigation
T1190	Exploit Public-Facing Application
T1059	Command and Scripting Interpreter
T1505.003	Server Software Component: Web Shell
T1005	Data from Local System
T1071.001	Application Layer Protocol: Web Protocols

These mappings describe plausible post-exploitation behaviors for defensive hunting. They are not confirmation that every technique has been observed in every PHP2Shell incident.

External Exposure Checks

Public tools can support initial triage, including the SOCRadar PHP2Shell Checker and Searchlight Cyber php2shell-scanner. Use them only against systems you own or are authorized to assess. Public scanners do not

replace authenticated asset inventory, package verification, vulnerability scanning or incident investigation. Searchlight Cyber notes that reliable active testing may intentionally trigger an out-of-memory condition and therefore carries operational risk; defenders should follow the tool author's cautions before using active probes in production.

Summary Conclusion

PHP2Shell should be treated as a critical, externally exploitable PHP risk with confirmed exploitation attempts in the threat landscape. The immediate objective is not merely to identify a website that "uses PHP"; it is to identify every web-serving or embedded PHP runtime, determine whether an effective fix is installed, reduce exposure for systems that cannot be fixed immediately, and investigate vulnerable internet-facing systems for signs of exploitation.

References

- **PHP Project security releases:** <https://www.php.net/archive/2026.php#2026-10-21-5>
- **NCSC-NL PHP2Shell operational repository:** <https://github.com/NCSC-NL/php2shell>
- **Sophos PHP2Shell advisory:** <https://www.sophos.com/en-us/security-advisories/sophos-sa-20261021-php2shell>
- **Palo Alto Networks Unit 42 threat brief:** <https://unit42.paloaltonetworks.com/cve-2026-109618-cve-2026-109619/>
- **Sansec PHP2Shell research:** <https://sansec.io/research/php2shell>
- **SOCRadar PHP2Shell Checker:** <https://socradar.io/free-tools/php2shell-checker>
- **SANS emergency PHP vulnerabilities resource:** <https://www.sans.org/go/php-vulnerabilities-what-you-need-know-today-do-right-now>

Need Assistance or Want to Discuss Further?

Contact Blackswan Cybersecurity at 855-BLK-SWAN or Contact@BlackswanCybersecurity.com