Hive Pro

Threat Level

Red

Hiveforce Labs

THREAT ADVISORY

VULNERABILITY REPORT

PHP2Shell: Your Application Code Is Clean, But the Engine Underneath It Isn't

Date of Publication	Last Update Date	Admiralty Code	TA Number
October 22, 2026	October 27, 2026	A1	TA2026309

Summary

First Seen: August 2026
Affected Products: PHP (all versions from 5.4 through 8.5.11)
Priority Targeted Regions: USA, Japan, Thailand, Indonesia, South Korea, Australia, Taiwan, India, Singapore, UAE, Saudi Arabia, and the EU
Priority Targeted Industries: Government, financial, logistics, airports, manufacturing, and online transaction platforms
Threat Actor: Linked to North Korea's Lazarus ecosystem
Impact: PHP has shipped emergency patches for two critical vulnerabilities, CVE-2026-109618 and CVE-2026-109619, tracked together as PHP2Shell. Both sit in the core PHP engine rather than in any application built on top of it, so a site can be fully up to date and still be exposed simply because it runs PHP. The flaws let an unauthenticated attacker run code on the server with the web worker's privileges and read raw memory from PHP worker processes, which can contain passwords, session tokens, API keys, and certificates from other users' requests. The flaws allow pre-authentication remote code execution against default WordPress, Drupal, and Joomla deployments, though exploitation does not depend on those platforms at all. Nearly every PHP release since 5.4 is affected by at least one of the two issues, putting most of the world's websites in scope. A Korean-speaking group assessed with moderate confidence to be linked to North Korea's Lazarus ecosystem is also reported to be preparing a targeted campaign, having identified 216 unpatched internet-facing hosts.

CVEs

CVE	NAME	AFFECTED PRODUCT	ZERO-DAY	CISA KEV	PATCH
CVE-2026-109618	PHP2Shell (PHP Memory Corruption Vulnerability)	PHP Group PHP	✗	✓	✓
CVE-2026-109619	PHP2Shell (PHP Stale Internal Request-State Vulnerability)	PHP Group PHP	✗	✓	✓

Vulnerability Details

#1

An out-of-band emergency release from the PHP security team on 21 October 2026 addressed two critical remote code execution flaws, CVE-2026-109618 and CVE-2026-109619, grouped under the name PHP2Shell. Both are memory-safety defects in the core engine, and both trace back to the same underlying behaviour. When a request ends abnormally, most often because PHP hits an out-of-memory condition, the engine can tear down the request's heap while internal bookkeeping still points at that freed memory or still holds data that was only partially set up. The worker process survives and goes on to handle the next request, carrying that corrupted state with it. Because PHP workers are reused across requests, an attacker who can reliably trigger the failure can line up their own data in the memory the engine is about to trust.

#2

CVE-2026-109618 is the out-of-memory corruption path. An out-of-memory error interrupts an operation partway through, leaving request data only partially initialized, and PHP's cleanup routines then process that invalid state as if it were valid. Chaining the weaknesses in this path gives remote code execution running as the web worker account. CVE-2026-109619 is the stale global state path, where PHP keeps references to request-owned data after the backing memory has been freed; a later request handled by the same worker reaches those references before normal initialization resets them, producing a heap use-after-free that has been shown to yield both memory disclosure and remote code execution. The confirmed routes into this second flaw run through session upload-progress handling and, on PHP 8.5, through retained fatal-error backtraces.

#3

Neither flaw requires authentication, and neither requires the target application to contain a bug of its own, which is what makes the attack surface so large. A minimal "Hello World" script on an affected runtime is exploitable. PHP has stated that the risk is highest for Apache mod_php, where a demonstrated exploit fires before any application code runs at all, and PHP-FPM is not exempt because session-lifecycle remote code execution has been validated against it under suitable application conditions. Every PHP version from 5.4 up to 8.5.11 is affected. The fixes are in PHP 8.5.12, 8.4.27, 8.3.36, and 8.2.35. PHP 8.1 and anything older is past end of life, so those branches get no full release at all, and the project has published limited source backports for 7.4, 8.0, and 8.1 instead.

#4

Both CVEs carry a CVSS v4.0 base score of 9.2, placing them in the Critical band, which reflects unauthenticated network-reachable exploitation leading to full compromise of confidentiality, integrity, and availability. The attack traffic against these flaws was observed before patches were publicly available.

#5

Separately, a Korean-speaking group assessed with moderate confidence as linked to North Korea's Lazarus ecosystem is preparing a campaign built around these flaws. The group has put together a list of 216 internet-facing hosts that still appear unpatched, and is quietly probing them from 21 compromised machines, keeping the traffic slow and low in volume so it does not stand out. The group dropped hosts already running a fixed PHP version from the list. The targeting focuses on government bodies and banks first, then logistics, airports, manufacturing, and online payment platforms, across the US, Japan, South Korea, India, Singapore, Australia, Taiwan, Thailand, Indonesia, the UAE, Saudi Arabia, and the EU. No attack date has been set, and no successful break-ins have been confirmed.

Vulnerabilities

CVE ID	AFFECTED PRODUCTS	AFFECTED CPE	CWE ID
CVE-2026-109618	PHP Group PHP (5.4 through 8.5.11)	cpe:2.3:a:php:php:*:*:*:*:*:*:*:*	CWE-1325 CWE-416 CWE-755 CWE-824
CVE-2026-109619	PHP Group PHP (5.4 through 8.5.11)	cpe:2.3:a:php:php:*:*:*:*:*:*:*:*	CWE-1325 CWE-416 CWE-755 CWE-824

Recommendations

Patch PHP Right Away: Update to PHP 8.5.12, 8.4.27, 8.3.36, or 8.2.35, whichever matches your branch. This is the only complete fix. PHP 8.2 itself reaches end of life on 31 December 2026, so patching to 8.2.35 buys roughly ten weeks; plan the move to 8.3 or later now. If you are on PHP 8.1 or older, those branches are past end of life and will not get a full release, so either pull a backported build from your Linux distribution or LTS vendor, apply the limited source backports PHP published for 7.4, 8.0, and 8.1, or move to a supported version.

Start With Your Internet-Facing mod_php Servers: PHP has flagged Apache mod_php as the highest-risk configuration because a demonstrated exploit can fire before any of your application code runs. Work outward from there to PHP-FPM, php-cgi, lsphp, FrankenPHP, and LiteSpeed deployments. Anything that serves web requests and is reachable from the internet should be at the front of the queue.

Apply Temporary Mitigations Only If You Cannot Patch Today: For CVE-2026-109618, set memory_limit=-1 in a way applications cannot override, such as php_admin_value in the FPM pool or Apache configuration, keep post_max_size low at 8M without uploads or 32M with them, and use container or cgroup limits as an availability boundary. Do not substitute RLIMIT_AS or RLIMIT_DATA, which PHP warns can themselves trigger the vulnerable allocation failure. For CVE-2026-109619, set session.upload_progress.enabled=0 before request parsing and restart every worker, and on PHP 8.5 also set fatal_error_backtraces=0. Treat all of these as stopgaps that close known paths, not as protection.

Hunt for Signs of Compromise: Because these flaws execute below the application layer, your application logs may show very little. Look instead for unusual multipart requests followed by memory-limit errors or repeated HTTP 500 responses, unexpectedly large or malformed PHPSESSID response cookies, and unexplained PHP worker churn. Because these flaws can disclose raw worker memory, treat credentials, session tokens, and API keys that lived on an exposed server as potentially compromised and rotate them.

Edge Blocking for CVE-2026-109618: Block form field names of 1,000 bytes or more. Every variant of this flaw needs a field name large enough that copying the key exhausts memory_limit during request startup, which in practice means megabytes, so a 1 KB threshold sits far below the exploit minimum and still produces no false positives. Match names and not values by anchoring at a field boundary on urlencoded bodies, and cover the multipart variant separately by checking part header name lengths. Never gate on Content-Length, because chunked requests do not carry one and HTTP/2 does not require one, which makes a header-based size check a one-line bypass. As a backstop, rate-limit large POSTs to roughly ten per source IP per minute.

Edge Blocking for CVE-2026-109619: Exploit bodies here run to roughly 450 bytes with short field names, so there is nothing to size-limit. Instead block three multipart shapes that no legitimate client sends, matching on part headers only so that uploaded file contents never trigger the rules. First, a PHPSESSID form field, since the upload handler copies it into the session ID without checking session.use_only_cookies and real clients send session IDs in the Cookie header. Second, a PHP open tag in a filename, limited to the full and short echo tags to avoid false positives on XML-style names. Third, serialized-object syntax in a filename, which is how the handler-mismatch chain writes its payload. Log the PHP_SESSION_UPLOAD_PROGRESS field but do not block on it, because that is the legitimate progress-bar mechanism. These rules stop the write to the session file but not a later local file inclusion read, so purge session.save_path as part of the rollout.

PHP Fatal Error From an Unknown Script: In Apache or Nginx error logs, look for messages containing PHP Fatal error together with “tried to allocate” and “in Unknown on line 0”. Ordinary application memory exhaustion names the file and the line responsible, so the Unknown on line 0 phrasing means the allocation happened outside normal script execution. Treat this as a high-priority lead rather than proof, because it is not unique to PHP2Shell, and correlate it with oversized POSTs, connection resets, and worker crashes from the same source.

Web Server Spawning a Shell: Look for process events where the parent is apache2, httpd, nginx, lighttpd, caddy, php-cgi, php-fcgi, php-fpm, lsphp, or frankenphp, or where the process runs as a web server account such as apache, www-data, or nginx, and the child process is a shell such as sh, bash, dash, zsh, or ksh. Narrow on command lines containing -c or -lc alongside suspicious content such as id, whoami, hostname, nc, ncat, socat, xxd, base64, openssl, curl, wget, chmod, python, perl, /dev/tcp, /dev/shm/, /tmp/, /.ssh, /etc/passwd, or /etc/shadow. Web servers do legitimately spawn shells in some environments, so this one needs tuning.

Post-Exploitation Hunting: The exploit runs shell commands as the web server user without crashing the worker, so there may be no visible disruption at all. Look for unexpected child processes of web server and PHP workers, files created or modified in /tmp, /dev/shm, /var/tmp, or web-writable directories by the web server user, and outbound connections from that user on non-standard ports. In logs, look for bursts of HTTP 200 responses to large POSTs from a single source carrying Allowed memory size errors, since each successful attempt returns a 200, and for hundreds of large POSTs from one address producing AH00052 child process exit entries in Apache error logs. For persistence, check new or modified SSH authorized_keys files under web server user home directories or /var/www, new cron entries for that user, and unfamiliar PHP files in web roots. Cover the period from 17 October 2026, the earliest recorded exploitation attempt, rather than from the disclosure date or the date you patched.

Potential MITRE ATT&CK TTPs

Tactic	Technique	Sub-technique
Initial Access	T1190: Exploit Public-Facing Application	
Execution	T1059: Command and Scripting Interpreter	T1059.004: Unix Shell

Tactic	Technique	Sub-technique
Persistence	T1505: Server Software Component	T1505.003: Web Shell
	T1098: Account Manipulation	T1098.004: SSH Authorized Keys
	T1053: Scheduled Task/Job	T1053.003: Cron
Discovery	T1033: System Owner/User Discovery	
Credential Access	T1212: Exploitation for Credential Access	
	T1552: Unsecured Credentials	T1552.001: Credentials In Files
Collection	T1005: Data from Local System	
Defense Evasion	T1211: Exploitation for Defense Evasion	
Reconnaissance	T1595: Active Scanning	T1595.002: Vulnerability Scanning
	T1590: Gather Victim Network Information	
Resource development	T1588: Obtain Capabilities	T1588.005: Exploits
	T1608: Stage Capabilities	
	T1583: Acquire Infrastructure	T1583.004: Server
Command and Control	T1105: Ingress Tool Transfer	
Lateral Movement	T1210: Exploitation of Remote Services	
Exfiltration	T1041: Exfiltration Over C2 Channel	
Impact	T1496: Resource Hijacking	
	T1486: Data Encrypted for Impact	

Indicators of Compromise (IOCs)

TYPE	VALUE
IPv4	103[.]216[.]220[.]92*, 31[.]223[.]184[.]166*, 37[.]19[.]200[.]5*, 134[.]209[.]159[.]64*, 103[.]114[.]163[.]188*, 95[.]217[.]98[.]37*, 5[.]191[.]140[.]240, 158[.]69[.]113[.]75, 94[.]154[.]43[.]149, 143[.]246[.]223[.]124, 45[.]198[.]224[.]169, 85[.]11[.]167[.]243, 118[.]112[.]100, 198[.]224[.]169, 175[.]189[.]76, 142[.]93[.]155[.]162, 134[.]122[.]38[.]73, 5[.]255[.]114[.]109, 165[.]22[.]232[.]198, 159[.]223[.]59[.]200, 192[.]177[.]131[.]13, 94[.]154[.]42[.]149, 173[.]249[.]16[.]178, 170[.]64[.]224[.]137, 170[.]64[.]226[.]40, 45[.]148[.]10[.]80, 152[.]237[.]230[.]29, 146[.]70[.]194[.]93, 185[.]157[.]109[.]60, 143[.]246[.]213[.]138, 144[.]91[.]70[.]51, 217[.]60[.]196[.]124, 151[.]253[.]243[.]219, 23[.]106[.]131[.]214, 173[.]249[.]16[.]192, 45[.]148[.]10[.]201, 146[.]70[.]114[.]180, 146[.]70[.]114[.]182, 147[.]175[.]149[.]67, 147[.]78[.]168[.]57,

Note: The * marked IPs are shared anonymization infrastructure; investigate any hits rather than blocking them.

TYPE	VALUE
IPv4	109[.]56[.]219[.]150, 158[.]173[.]75[.]178, 185[.]203[.]18[.]54, 185[.]235[.]212[.]105, 103[.]77[.]210[.]80, 128[.]90[.]176[.]13, 138[.]199[.]34[.]201, 145[.]255[.]150[.]25, 149[.]102[.]232[.]44, 149[.]102[.]232[.]93, 158[.]173[.]242[.]133, 147[.]78[.]168[.]67, 102[.]207[.]250[.]188, 94[.]154[.]43[.]185, 75[.]119[.]143[.]189, 117[.]203[.]73[.]63, 146[.]190[.]107[.]103, 95[.]217[.]164[.]124, 124[.]122[.]142[.]142, 188[.]80[.]228[.]122, 134[.]209[.]181[.]133, 134[.]209[.]252[.]157, 119[.]63[.]72[.]54, 138[.]68[.]161[.]170, 95[.]216[.]5[.]90, 103[.]21[.]161[.]105, 115[.]96[.]27[.]126, 185[.]207[.]232[.]121, 78[.]47[.]85[.]99, 92[.]118[.]112[.]100
URLs	hxxp[:]//92[.]118[.]112[.]100/nfs-setup, hxxp[:]//92[.]118[.]112[.]100/mountpoint
SHA256	e2529d4afa39ae4db2eb05d5176317c906289abf09660c75bdeec43e54c55aa3, 0c5486f06620e26e316f4b33f318b0d646d5e1133789331aa2fed012360d1486, 267a3c8ed021a0a52c29bf076617a0dfd5cd5285202b8d0bd9ad0712110ccae8
MD5	b61a098fc62a2c2696e196dca78da71a

Patch Link

https://www.php.net/archive/2026.php#2026-10-21-5

References

https://www.wiz.io/blog/php2shell-php-rce

https://www.php.net/archive/2026.php#2026-10-21-5

https://cloud.google.com/blog/products/identity-security/responding-to-core-php-vulnerabilities-cve-2026-109618-cve-2026-109619

https://www.abstract.security/blog/php2shell-critical-php-rce-vulnerabilities-cve-2026-109618-and-cve-2026-109619

https://zerodayclock.com/explorer?cve=CVE-2026-109618

https://zerodayclock.com/explorer?cve=CVE-2026-109619

https://calif.io/php2shell/#php-5-4-0-rce

https://aws.amazon.com/blogs/security/amazon-threat-intelligence-identifies-php2shell-exploitation-prior-to-public-disclosure-in-madpot/

https://www.cyfirma.com/research/cyfirma-threat-advisory-suspected-dprk-campaign-staging-php2shell-targeting-against-global-php-infrastructure/

What Next?

At **Hive Pro**, it is our mission to detect the most likely threats to your organization and to help you prevent them from happening.

Book a Demo of **HivePro**.

REPORT GENERATED ON

October 22, 2026 • 11:00 AM

© 2026 All Rights are Reserved by Hive Pro

More at www.hivepro.com