

THREAT HUNTING

LAB

PHP2Shell: Cross-Request Use-After-Free in PHP Core enables Pre-Auth RCE

Global Weekly Threat Overview

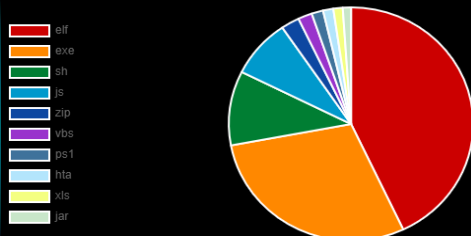
Global Weekly Notable One

Threat Hunting Activity

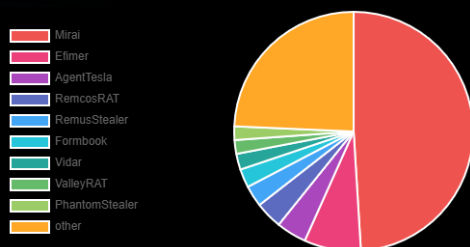
Global Weekly Threat Overview

Critical GitLab Flaw (CVSS 10.0) Spans 12 Years of Releases. GitLab shipped fixes on October 21 for CVE-2026-110264, an integer overflow in the Ruby Onigmo regex engine that GitLab bundles and patches in its own builds. Under conditions GitLab has not disclosed, an unauthenticated user could run arbitrary code on self-managed instances, with every version back to 7.0 (2014) affected. Notably, administrators who patched on October 12 are exposed again: that update only covered 19.2 and later, and the new fix sits one patch above it. GitLab.com and GitLab Dedicated already run fixed builds. No in-the-wild exploitation confirmed as of October 22, and the flaw is not yet in CISA's KEV catalog. GitLab also published two Sigma detection rules via its open-source TLDR framework.

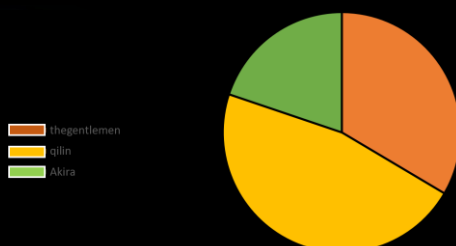
Top 10 file types



Top 10 malware family



Top 3 Ransomware Group



Tailworm: Tailscale SSH Lets a Compromised Device Reach Root Elsewhere. Researcher Erica Windisch (Cyberdione Labs) published "Tailworm" on October 25, showing that a single compromised device on a Tailscale network, with SSH and port forwarding enabled, can obtain root on other reachable servers without stealing any password or key, since Tailscale SSH authorizes by device identity rather than by user credential. Tested against Tailscale 1.104.2 and self-hosted headscale 0.29.5. No CVE or security bulletin existed at publication; Tailscale confirmed it is preparing a client-side fix. On headscale, forwarding cannot be disabled at all, so every SSH grant there should be treated as granting root.



Initial Access: Exploit Public-Facing Application

PHP2Shell groups four related memory-safety bugs under two critical CVEs (CVE-2026-109618 and CVE-2026-109619, both CVSS v4.0 9.2) in the PHP engine itself, the most widely deployed server-side web runtime. Unlike typical application vulnerabilities, these flaws sit below any application code. A PHP worker reuses its memory across requests, and when a request exceeds 'memory_limit', PHP destroys the request heap on a fatal out-of-memory error but fails to clear global pointers into that heap. A subsequent request handled by the same worker can then read or act on stale pointers into freed memory.

The practical impact ranges from silent cross-user memory disclosure (credentials, session tokens, API keys, even TLS private keys leaked from other requests handled by the same worker) to full unauthenticated remote code execution with no login, application bug or user interaction required. Because the vulnerable code lives in the engine rather than in any specific product, even a minimal, cleanly written PHP application on an affected version remains exposed, and popular platforms built on PHP inherit the exposure regardless of their own code quality.

Threat Hunting Activity

TACTIC

Initial Access

TECHNIQUES

T1190 - Exploit Public-Facing Application

Adversaries may attempt to exploit a weakness in an Internet-facing host or system to initially access a network. The weakness in the system can be a software bug, a temporary glitch, or a misconfiguration. Exploited applications are often websites/web servers, but can also include databases (like SQL), standard services (like SMB or SSH), network device administration and management protocols (like SNMP and Smart Install), and any other system with Internet-accessible open sockets.

Threat Hunting Activity

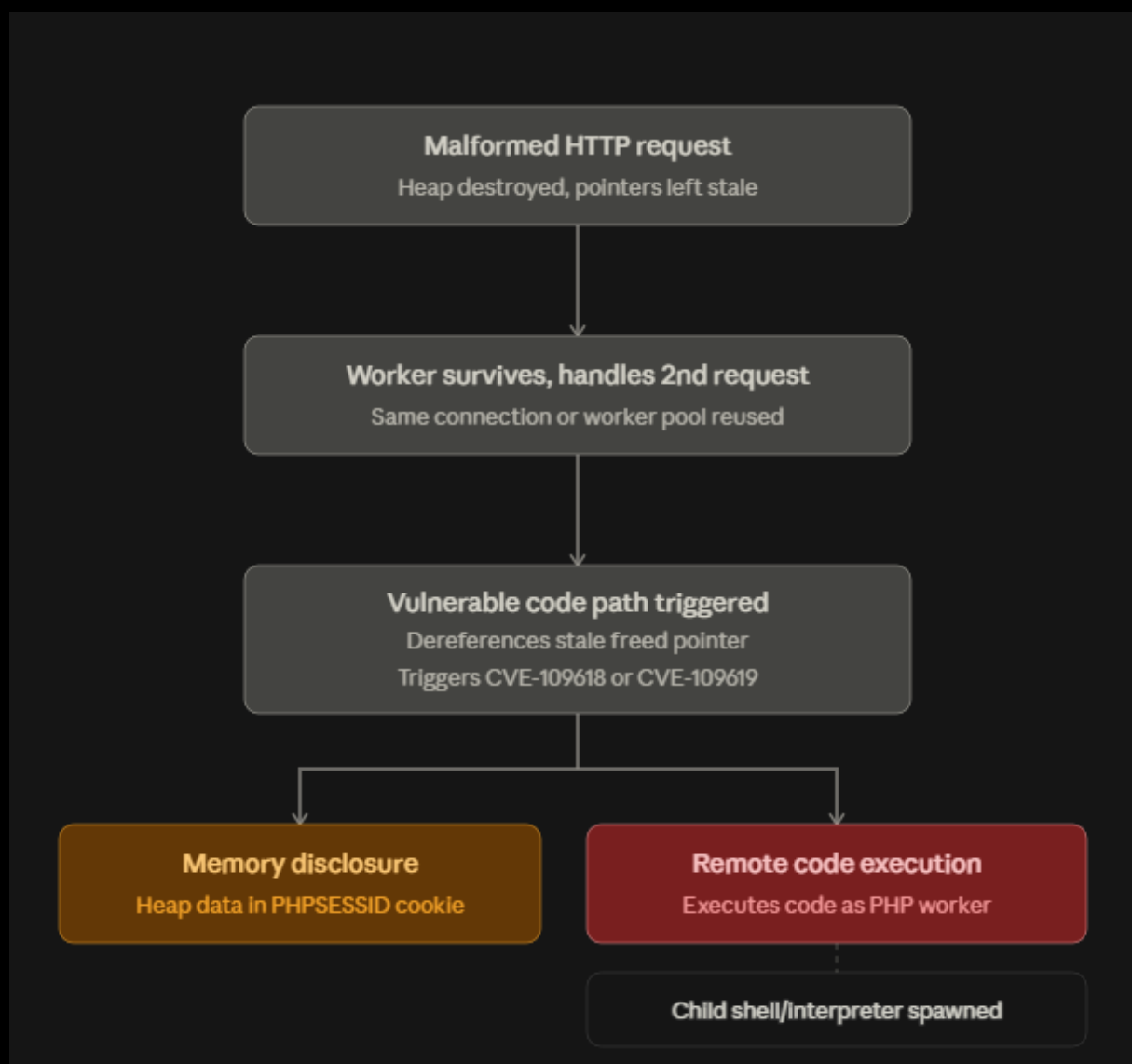
The shared root cause is a lifecycle mismatch between PHP's per-request memory model and its persistent worker processes. Each PHP worker normally resets its internal state between requests, but a fatal out-of-memory error short-circuits that reset: the request heap is destroyed immediately, while global pointers that reference objects inside that heap are left untouched. The next request picked up by the same worker then operates on those dangling references as if they were valid.

'CVE-2026-109618' centers on session upload progress, a feature enabled by default that tracks multipart/form-data upload status in global unserializer state, and runs during request parsing before any application code, even without a call to 'session_start()'. An out-of-memory condition during this phase leaves that global state "published" against a destroyed heap. A related path also leaks the session ID held in a global pointer: a later multipart request on the same worker reaches the upload-progress cleanup routine before session initialization resets the pointer, causing PHP to URL-encode the reclaimed heap content directly into the 'PHPSESSID' response cookie sent back to the client, exposing fragments of other users' request memory without needing RCE at all.

'CVE-2026-109619' covers two related primitives. The first is a use-after-free in fatal-error backtrace handling: PHP retains the last fatal error's backtrace in a global, which an out-of-memory event does not clear, so a subsequent request on the worker finds a pointer into the prior request's freed memory. The second is OPcache type confusion: while OPcache persists a compiled script into shared memory, an allocation failure at the wrong moment leaves internal slots inconsistent across workers, causing one worker to later interpret shared data under the wrong type, corrupting shared PHP state in a way that has been developed into working RCE chains against common CMS platforms.

Threat Hunting Activity

Exploitation requires no authentication and no prior access, only the ability to send crafted HTTP requests capable of triggering an out-of-memory condition followed by a second request on the same worker. Vulnerable PHP versions span 5.4 through 8.1 without official fixes (8.1 and earlier require vendor backports or an upgrade), while 8.2.35, 8.3.36, 8.4.27 and 8.5.12 contain the fix.



Threat Hunting Activity

Detection can be built on Event ID 4688 looking for a PHP worker process ('php-cgi.exe', 'php.exe', or the hosting 'w3wp.exe'/'httpd.exe') spawning an unexpected child interpreter such as 'cmd.exe' or 'powershell.exe', correlated on Linux with auditd execve events showing 'php-fpm'/'apache2'/'nginx' worker processes forking a shell or network utility not expected for that application's normal behavior.

Date and time	Event ID	Task category	New process name	Creator process name
10/27/2026 11:42:07	4688	Process creation	cmd.exe	php-cgi.exe
10/27/2026 11:41:52	4688	Process creation	powershell.exe	w3wp.exe
10/27/2026 11:40:18	4688	Process creation	cmd.exe	php.exe
General				
New process name		C:\Windows\System32\cmd.exe		
Creator process name		C:\xampp\php\php-cgi.exe		
Process command line		cmd.exe /c whoami		
Account name		IIS APPPOOL\DefaultAppPool		



THREAT HUNTING



SORINT_{SEC}