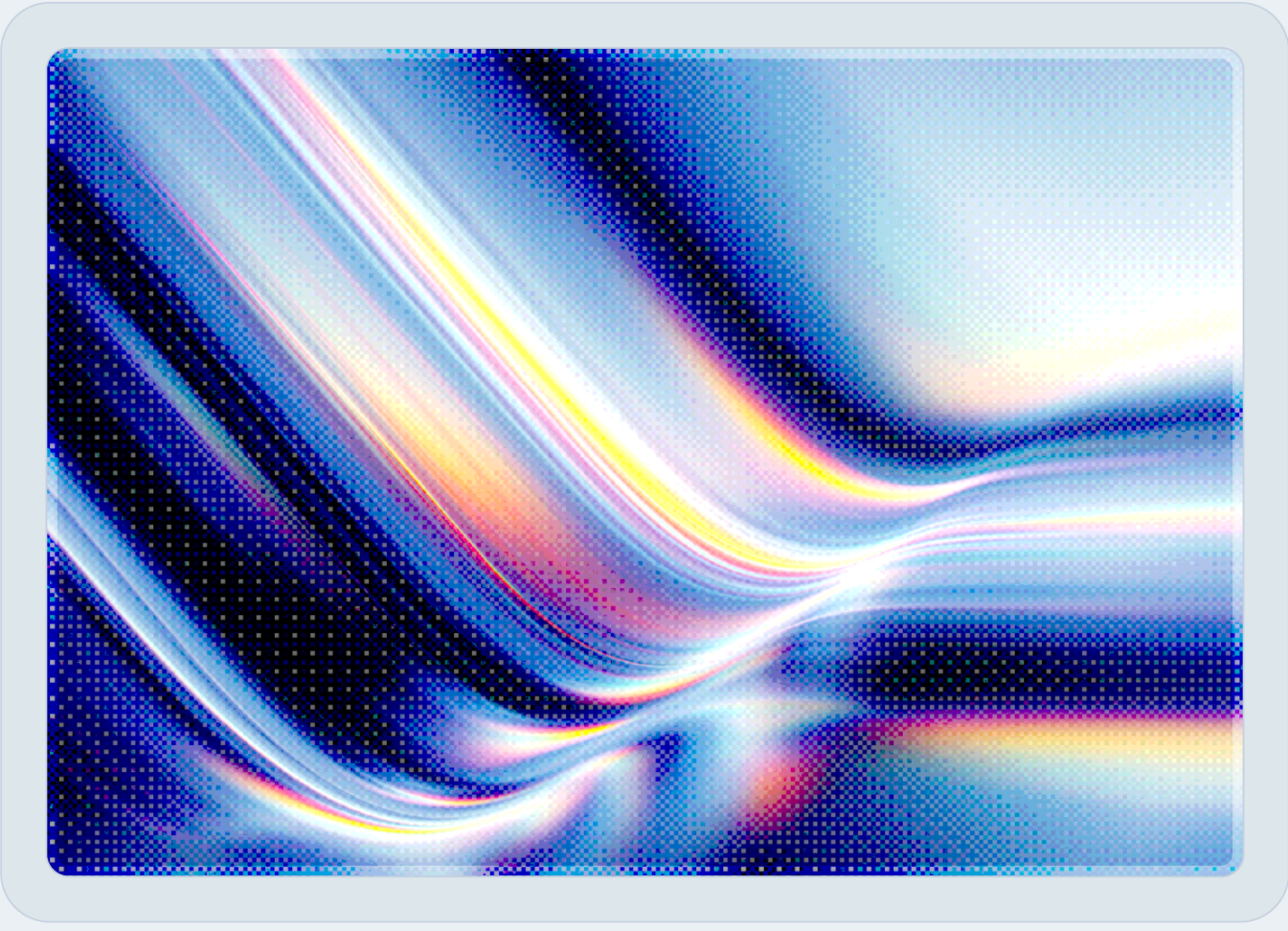
CSAI Foundation | Cloud Security Alliance

Upload2Shell: AI-Assisted Ghostscript Sandbox Escape and Distribution Patch Lag

CVE-2026-125396, file-upload pipelines, and the gap between upstream fixes and enterprise packages

2026-11-23

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

© 2026 Cloud Security Alliance. Some rights reserved.

You may download, store, display, view, print, redistribute, and link to this document in its original, unmodified form, provided that attribution to the Cloud Security Alliance is maintained and all trademark and copyright notices remain intact.

This document may not be modified or altered. You may quote portions of the document as permitted by the Fair Use provisions of the United States Copyright Act, provided that attribution is given to the Cloud Security Alliance.

This document may be shared on professional and social media platforms in its original form with attribution.

This document was generated with AI assistance and has not undergone official CSA review and approval processes.

Key Takeaways

CVE-2026-125396, which its discoverers call "Upload2Shell," is a use-after-free in Ghostscript's Type 1 font-loading code. A crafted document can use it to disable the `-dSAFER` sandbox and run attacker-chosen commands on the server that renders the file [1][4]. The researcher who disclosed it, Paulos Yibelo of pwn.ai, says the flaw was found by "GPT 5.6 Sol, working with" him [1]. Artifex fixed the bug in Ghostscript 10.08.0, released on 8 September 2026 [7]. The CVE record was not published until 9–10 November, roughly two months later [2][5][8].

The practical exposure comes from how files reach Ghostscript, not from the bug class itself. Image and document libraries such as ImageMagick, Pillow, libvips, and a Qt image plugin pass PostScript or EPS content to Ghostscript even when the file is named `.png`, `.jpg`, or `.pdf`. Through these routes pwn.ai reports command execution in ten application configurations, including Debian 13 CUPS, Roundcube, MediaWiki, Plesk, and WordPress Core, although in one Plesk test an enforced AppArmor profile blocked the shell [1][2][3]. CSA has not reproduced these results. Several of these paths required no authentication [3].

The patch lag is the part that should concern enterprise defenders most. As of pwn.ai's 12 November survey, Amazon Linux 2, 2023, and 2027 Preview showed "Pending Fix", several SUSE Linux Enterprise lines were "in progress", and the status of the Red Hat Enterprise Linux family could not be established [1]. Amazon's tracker and Red Hat's bug record showed no fix as of this writing [5][6]. CVE-keyed scanners had no identifier to match for the two months between the upstream release and CVE publication. At least one distribution advisory that shipped the fix during that window listed it without a CVE number [9].

No public source reports exploitation in the wild, and the CVE is not in CISA's Known Exploited Vulnerabilities catalog [3][8]. However, a third party has posted a comment on Red Hat's bug tracker pointing to a public GitHub proof of concept with reverse-shell payloads. CSA has not verified that claim [6].

Organizations should inventory every Ghostscript instance, including copies inside container images, and confirm each runs 10.08.0 or a vendor backport. Where no fixed package exists, they should disable PostScript and EPS handling in upload pipelines or isolate the rendering. They should also treat the upstream fix date, not the CVE date, as the start of the exposure clock for dependencies that sit under file-upload paths.

Background

Ghostscript is Artifex Software's interpreter for PostScript and PDF. It has been part of Linux server stacks for decades, often without application owners being aware of it. Print systems such as CUPS use it to convert jobs. Image-processing libraries call it whenever they meet PostScript or Encapsulated PostScript (EPS) data. Document management, wiki, webmail, and content management platforms reach it indirectly whenever they produce a thumbnail or preview. Because PostScript is a full programming language, Ghostscript ships a sandbox mode, `-dSAFER`, which restricts file access and blocks dangerous operators. Applications that render untrusted input are expected to run with that sandbox on. Ghostscript has a long history of `-dSAFER` bypasses. When the fix landed, Artifex developer Ken Sharp described the vulnerable code as "an ancient piece of code well known to be a vulnerability, but difficult to eradicate" [3].

The Upload2Shell flaw sits in `.putgstringcopy`, a helper procedure in `Resource/Init/gs_fonts.ps`. The helper uses the privileged internal operator `.forceput` to write into protected font dictionaries, and it did not check operand types before doing so [1][4]. According to pwn.ai's write-up, a crafted Type 1 font forces a type error inside the helper. A custom error handler then resumes execution at `.forceput` without ever calling the hidden operator directly. A save/restore cycle leaves a dangling reference that the attacker uses as a write-what-where primitive [1]. The researchers used that primitive to zero the in-memory flag that enforces path controls, widen file-writing permissions, and unlock file permissions. They then executed a command through Ghostscript's `%pipe%` device. As the write-up puts it, "`-dSAFER` was still on the Ghostscript command line; its native file-path check had been disabled in memory" [1]. The public proof of concept from pwn.ai shows only the memory-corruption step (a function-pointer swap), not shell execution, and it fails on 10.08.0 [1] [3].

The disclosure timeline shows how long the window between discovery and broad availability of a fix can be, even when the upstream maintainer moves quickly. The table below combines pwn.ai's published timeline with the vendor records that CSA checked independently.

Date (2026)	Event	Source
27 July	Flaw discovered with AI assistance	[2]
12 August	WordPress 7.0.4 closes its own path (separate CVE-2026-65640)	[2][3]

Date (2026)	Event	Source
By 2 September	Reported upstream as Artifex Bug 709684	[2]
8 September	Ghostscript 10.08.0 released with fix	[1][7]
12 September	10.08.0 enters Debian unstable	[2]
25 September	Debian 13 security update (DSA-6516-1)	[2][4]
21 October	Ubuntu USN-8949-1 ships fixes, listing this issue without a CVE number	[2][9]
6 November	CVE requested	[2]
9–10 November	CVE-2026-125396 published	[2][5] [8]
12 November	pwn.ai distribution survey; Amazon tracker last modified	[2][5]
21 November	Full technical write-up published	[1]

Severity scoring for the CVE is not consistent across sources. Amazon's advisory and TheHackerWire list a CVSS v3.1 base score of 7.8 with a local attack vector and user interaction required [5][8]. pwn.ai and The Hacker News, citing VulnCheck, give 8.5 under CVSS 4.0 [1][3]. The local attack vector follows the CVSS convention for file-parsing flaws, which assumes a user opens the file. pwn.ai argues that its demonstrations show remote, unauthenticated delivery through upload and print features [1]. CSA considers the delivery evidence more relevant to prioritization than the base score.

Security Analysis

The delivery problem: Ghostscript behind the upload button

The finding matters because applications rarely call Ghostscript directly. pwn.ai's central observation is that "the upload endpoint does not have to start `gs` itself" [1]. Content sniffing in common libraries sends PostScript or EPS bytes to Ghostscript based on what the file contains, not what it is called. In the

researchers' testing, Pillow passed EPS content to Ghostscript when an application called `load()`, `crop()`, or `thumbnail()`. PHP's Imagick extension did so through ImageMagick's PostScript delegate. libvips, pyvips, and Ruby Vips did so through `magickload` in builds that include ImageMagick. The Qt KImageFormats EPS plugin did so through `QImageReader.read()` [1]. Opening a file with `Image.open()` alone did not trigger rendering in Pillow, which shows how much the outcome depends on the exact code path [1]. Several stacks did not start Ghostscript in testing at all, including PHP GD, OpenCV, prebuilt Sharp, Jimp, Java ImageIO, and Go's standard decoders [1].

This indirection undermines two defensive assumptions. The first is that searching application code for `gs` invocations reveals exposure; it does not, because the call happens inside a library and sometimes in-process [1]. The second is that extension allow-lists protect upload endpoints. A file named `quarterly-results.jpg` and served as `image/jpeg` reached Ghostscript through Roundcube's attachment thumbnailing [1].

Demonstrated reach

The table below summarizes the application configurations in which pwn.ai reports demonstrated command execution. CSA has not reproduced these results. Versions and preconditions are as the researchers state them.

Application (tested version)	Access required	Path to Ghostscript	Source
Debian 13 CUPS	Anonymous IPP print job to a shared queue	`gstopdf` filter	[1][3]
Roundcube 1.7.2	Recipient views email; no click	Attachment thumbnail via ImageMagick (permissive delegate)	[1][2]
WordPress 7.0.2 + Forminator 1.55.1	Unauthenticated form upload	Image processing	[1][3]
WordPress 7.0.2 + Essential Addons for Elementor Lite 6.7.0	Unauthenticated form upload	Image processing	[1][3]

Application (tested version)	Access required	Path to Ghostscript	Source
WordPress Core (7.0.0–7.0.3, 6.9.0–6.9.6)	Author role or higher	PDF thumbnailing	[1][3]
MediaWiki 1.46.0 + PdfHandler	Anonymous uploads enabled	PDF handling	[1]
DokuWiki 2026-07-14a	Optional ImageMagick backend	Image conversion	[1]
Mayan EDMS 4.11.5	POP3 email source	Document ingestion	[1]
ResourceSpace 11.0	Authenticated or upload-share	Preview generation	[1]
Plesk Obsidian 18.0.79	Webmail (Roundcube 1.6.17, GS 9.54.0)	Thumbnailing; AppArmor blocked shell in one enforced test	[1][2]

The researchers also traced code paths, without demonstrating execution, in Gradio and through source review in InvokeAI, AUTOMATIC1111 Stable Diffusion WebUI, ComfyUI, Frappe/ERPNext, Home Assistant Core, CVAT, Paperless-ngx, and django-filer [1]. Several of these are AI image-generation, annotation, or data-labeling tools. This suggests that machine-learning tooling, such as image-generation, annotation, and data-labeling applications, may inherit the same exposure as older content platforms, although pwn.ai did not demonstrate execution in any of these tools. pwn.ai further reports that the same CVE reached Google production upload paths, where only a `/proc/version` read was shown, and that Google mitigated that path on 10 November [1][2]. Reports involving Dropbox and other services concern separate Ghostscript bugs [2][3].

The patch lag in enterprise distributions

Upstream remediation was fast. Artifex wrote, committed, and released the fix no more than about a week after the date by which pwn.ai says it had reported the bug, and followed it with a change that moves the helper into C and removes its use of `.forceput` [1]. Rolling distributions and package channels picked up 10.08.0 within days. pwn.ai records Arch on 11 September, Debian unstable on 12 September, and Fedora Rawhide on 15 September [1][2].

Enterprise distributions moved more slowly. The table below shows status as recorded in pwn.ai's 12 November survey [1], updated with vendor trackers that CSA consulted on 23 November.

Distribution	Status 12 Nov (pwn.ai)	Status per vendor tracker, 23 Nov	Source
Debian 13 (trixie)	Fixed (DSA-6516-1, 25 Sep)	Fixed, 10.05.1~dfsg-1+deb13u3	[2][4]
Debian 12 (bookworm)	Vulnerable	Fixed in bookworm security via DLA-4914-1 (deb12u9)	[1][4]
Ubuntu 16.04–26.10	Fixed (USN-8949-1, 21 Oct)	Fixed	[2][9]
Amazon Linux 2 / 2023 / 2027 Preview	Pending Fix	Pending Fix	[1][5]
SLE 12 SP5 LTSS, selected SLE 15 SP4–SP7, SLES 16.0/16.1	In progress	Not independently checked	[1]
openSUSE Leap 16.0/16.1	Affected	Not independently checked	[1]
RHEL, Rocky, Alma, Oracle Linux, stable Fedora	Not established	Red Hat bug status NEW; no product fix listed	[1][6]

Two features of this pattern deserve attention. First, the CVE was published 62 days after the upstream release. During that period, vulnerability scanners and software composition analysis tools that key on CVE identifiers had nothing to flag, even though a fixed version existed and a public teaser claiming pre-authentication RCE against major platforms had circulated since 25 September [2]. Ubuntu's fix, which shipped before the CVE existed, was recorded against this issue without a CVE number [9]. As a result, a CVE-driven compliance process could show Ubuntu hosts as having no relevant finding before 9 November and as "remediated" after it, without ever registering the period of exposure. Second, widely deployed enterprise server distributions had not shipped a fix two weeks after CVE publication. Amazon Linux still listed the fix as pending [5], and Red Hat's tracker showed the bug in NEW status with no product fix or affectedness determination [6]. Organizations on Amazon Linux should treat themselves as exposed until a backport ships. RHEL-family operators should verify the installed Ghostscript version and, until Red Hat publishes a determination, apply the compensating controls described below as a precaution.

Container images add a third layer of lag. pwn.ai notes that "updating the host operating system does not update a separate copy" of Ghostscript bundled in an application or image [1][3]. A patched host running a container built months ago on an affected base image remains exposed. This is a plausible configuration for the document-processing and AI tooling listed above.

What AI-assisted discovery changes

The researchers' own account says only that a GPT model, working with a human researcher, found the flaw. They do not describe how the model was used [1]. CSA therefore does not draw conclusions about the specific workflow. The broader pattern is consistent with CSA's earlier analyses. Those analyses describe AI-assisted tooling finding real bugs in mature, heavily scrutinized codebases faster than downstream remediation and disclosure infrastructure can absorb them [10][11][14]. Upload2Shell is a concrete example in which the upstream maintainer was not the bottleneck. The delay sat in CVE assignment, distribution backporting, and the visibility of the fix to enterprise tooling. The roughly five weeks between discovery and the upstream report, during which WordPress fixed its own exposure, are not explained in pwn.ai's timeline [2]. As more parties point capable models at long-lived parsers and interpreters, defenders may face more cases in which a fix exists upstream well before any identifier, advisory, or vendor package reaches them. That inference rests on one case and on the trend evidence in CSA's prior work, not on measured data about lag intervals.

Recommendations

Immediate Actions

Vulnerability management teams should inventory every Ghostscript installation, including shared libraries linked into applications and copies inside container images, CI build images, and virtual appliances. They should confirm that each is version 10.08.0 or later, or a vendor backport that carries the `.putgstringcopy` fix (upstream commit `1484854e`) [4]. Teams should then identify which of those instances sit behind a path that accepts untrusted content, such as file uploads, email attachment previews, print queues, document ingestion, and wiki or CMS media handling. Those instances should be prioritized regardless of the CVSS base score. On Amazon Linux, RHEL-family, and SUSE hosts where no fixed package is yet available, apply compensating controls now rather than waiting for the vendor.

The most direct compensating control is to stop untrusted content from reaching Ghostscript. CSA recommends disabling the PS, EPS, EPI, PS2, PS3, and XPS coders in ImageMagick through `policy.xml`, and PDF where business needs allow. Where Pillow is used, avoid triggering EPS

rendering on untrusted files. Validate a file's real type by its content before any thumbnail or preview is generated, and reject PostScript where it is not an expected input. Several of the demonstrated paths used files presented as PDF, so where PDF coders must stay enabled, content-based type validation and isolated rendering are essential rather than optional [1]. Applications named in pwn.ai's research should be updated to vendor fixes, including WordPress 7.0.4 or 6.9.7 and Roundcube's CVE-2026-114727 release. CUPS queues exposed to a network should require authentication [2][3].

Short-Term Mitigations

Rendering of untrusted documents should run in an isolated, unprivileged sandbox with no network egress and no shell. Examples include a dedicated container with a seccomp or AppArmor profile, or a separate conversion service. pwn.ai reports that an enforced AppArmor profile blocked shell execution in one Plesk test [2]. That suggests mandatory access control can turn a sandbox escape into a contained failure, although a single test is not evidence of general effectiveness. Rebuild container images that bundle Ghostscript or ImageMagick from current base images, and add Ghostscript to the list of components that software composition analysis tools track by version as well as by CVE.

Vulnerability programs should also begin tracking upstream fix dates for a short list of high-exposure parsers and interpreters. Ghostscript, ImageMagick, and other libraries that handle untrusted file formats are reasonable starting points. Monitoring upstream release notes and security trackers narrows the window in which a fix exists but no CVE does, though upstream notes are often deliberately vague. Ghostscript's 10.08.0 notes, published two months before this CVE, said only that the release "addresses a number of potential security issues" [7]. The practical consequence is to treat any security-flagged release of a high-exposure interpreter as urgent, without waiting for details.

Strategic Considerations

In this case, CVE publication trailed the upstream fix by 62 days, so a CVE-keyed process would have lagged real exposure. Whether such gaps grow as AI-assisted discovery raises report volume is not yet clear. A recent systematization of AI vulnerability research found that disclosed vulnerabilities more than doubled in 2026, but that the intervals from report to fix and from disclosure to attack had "not shifted noticeably so far" [15]. Security leaders should therefore consider service-level expectations for distribution and container-base vendors that are measured from the upstream fix date. Procurement and vendor-risk processes should ask how quickly a vendor backports fixes for security-relevant components it redistributes, and how it signals fixes that do not yet have CVE identifiers.

Architecture teams should treat file-format conversion as a high-risk capability that deserves the same isolation as code execution, because rendering a PostScript-family file means executing a program supplied by the attacker. Converting the riskiest formats to safe representations in a separate tier, or avoiding PostScript-family support entirely, reduces dependence on how quickly each downstream vendor patches. Organizations deploying AI image-generation, annotation, and document-AI tooling should include those tools in this review, since several appear among the code paths pwn.ai traced [1].

CSA Resource Alignment

CSA's analysis *Project Glasswing: AI Discovery Outpaces Open Source Patching Capacity* documented AI-driven discovery producing more than 10,000 high- and critical-severity candidate findings. It also reported that only 97 of 1,596 disclosed vulnerabilities had been patched at the time of its analysis [10]. Upload2Shell shows a different stage of the same problem. Here the upstream maintainer patched within about a week of the report, and the bottleneck moved downstream to CVE assignment and enterprise distribution backports. That paper's recommendations on prioritizing high-exposure open-source dependencies and building direct maintainer channels apply directly to Ghostscript and similar interpreters. The CSA white paper *The Disclosure Bottleneck: When AI Discovery Outpaces Patch Consumption* argues that the bottleneck in software security has moved downstream to verification, disclosure, and patch consumption [14], and Upload2Shell is a concrete instance of that thesis. CSA's *AI-Accelerated Vulnerability Discovery and the Patch Debt Crisis* frames the resulting backlog as accumulated patch debt, which is the condition this note documents for enterprise Ghostscript packages [16].

The CSA research note *When Discovery Outruns Remediation: Open-Source Disclosure Strain* argues that enterprises should treat open-source maintainer capacity, and the intake and disclosure channels around it, as a supply-chain risk factor. It notes that public data on discovery-to-fix intervals for individual projects is limited [11]. The Upload2Shell timeline adds a documented interval: about six weeks from discovery to an upstream release, 62 days from release to CVE publication, and more than two months from release with no enterprise package on some major distributions. The note's recommendations on tracking maintainer and disclosure-channel health extend naturally to tracking distribution backport latency.

CSA's *AI Controls Matrix (AICM) v1.1* provides the control framing for operationalizing these recommendations [12]. Its Threat and Vulnerability Management domain covers inventory-driven vulnerability identification and remediation timelines, which this case suggests should be anchored to upstream fix dates as well as CVE dates. Its Application and Interface Security domain covers input validation and secure handling of untrusted content, which maps to content-type validation and isolation of document rendering. Its supply chain controls cover vendor patch commitments for redistributed

components. Organizations running AI tooling that processes user-supplied images and documents can use the AICM to assess whether those pipelines are covered by the same controls as conventional web applications.

The CSA *Frontier Ready* daily briefing for 22 November 2026 covered this disclosure and recorded the evidence status of each claim, including which statements are vendor-reported and which are CSA characterization [13].

References

[1] Paulos Yibelo, pwn.ai. "How We Hacked the Pipes — Upload2Shell." pwn.ai, 21 November 2026.

[2] Paulos Yibelo, pwn.ai. "Upload2Shell: Disclosure Timeline, Affected Applications, and Distribution Status." pwn.ai, updated 18 November 2026.

[3] Swati Khandelwal. "Researchers Demonstrate Ghostscript Sandbox-Escape RCE via Crafted File Uploads in Popular Apps." The Hacker News, 22 November 2026.

[4] Debian Security Team. "CVE-2026-125396." Debian Security Bug Tracker, accessed 23 November 2026.

[5] Amazon Web Services. "CVE-2026-125396." Amazon Linux Security Center, modified 12 November 2026.

[6] Red Hat. "Bug 2569593 — CVE-2026-125396 ghostscript: Arbitrary code execution via SAFER sandbox bypass." Red Hat Bugzilla, accessed 23 November 2026.

[7] Artifex Software. "Ghostscript News — Version 10.08.0 (2026-09-08)." Ghostscript Documentation, 8 September 2026.

[8] TheHackerWire. "CVE-2026-125396." TheHackerWire Vulnerability Database, accessed 23 November 2026.

[9] Canonical. "USN-8949-1: Ghostscript vulnerabilities." Ubuntu Security Notices, 21 October 2026.

[10] Cloud Security Alliance. "Project Glasswing: AI Discovery Outpaces Open Source Patching Capacity." CSA, June 2026.

[11] Cloud Security Alliance. "When Discovery Outruns Remediation: Open-Source Disclosure Strain." CSA Labs, October 2026.

[12] Cloud Security Alliance. "AI Controls Matrix (AICM) v1.1." CSA, June 2026.

[13] Cloud Security Alliance. "Frontier Ready Daily — 22 November 2026." CSA Labs, 22 November 2026.

[14] Cloud Security Alliance. "The Disclosure Bottleneck: When AI Discovery Outpaces Patch Consumption." CSA Labs, 6 November 2026.

[15] arXiv. "SoK: Assessing the Evolution of LLM Capability in Software Vulnerability Research." arXiv:2611.26601, 19 November 2026.

[16] Cloud Security Alliance. "AI-Accelerated Vulnerability Discovery and the Patch Debt Crisis." CSA Labs, 2026.