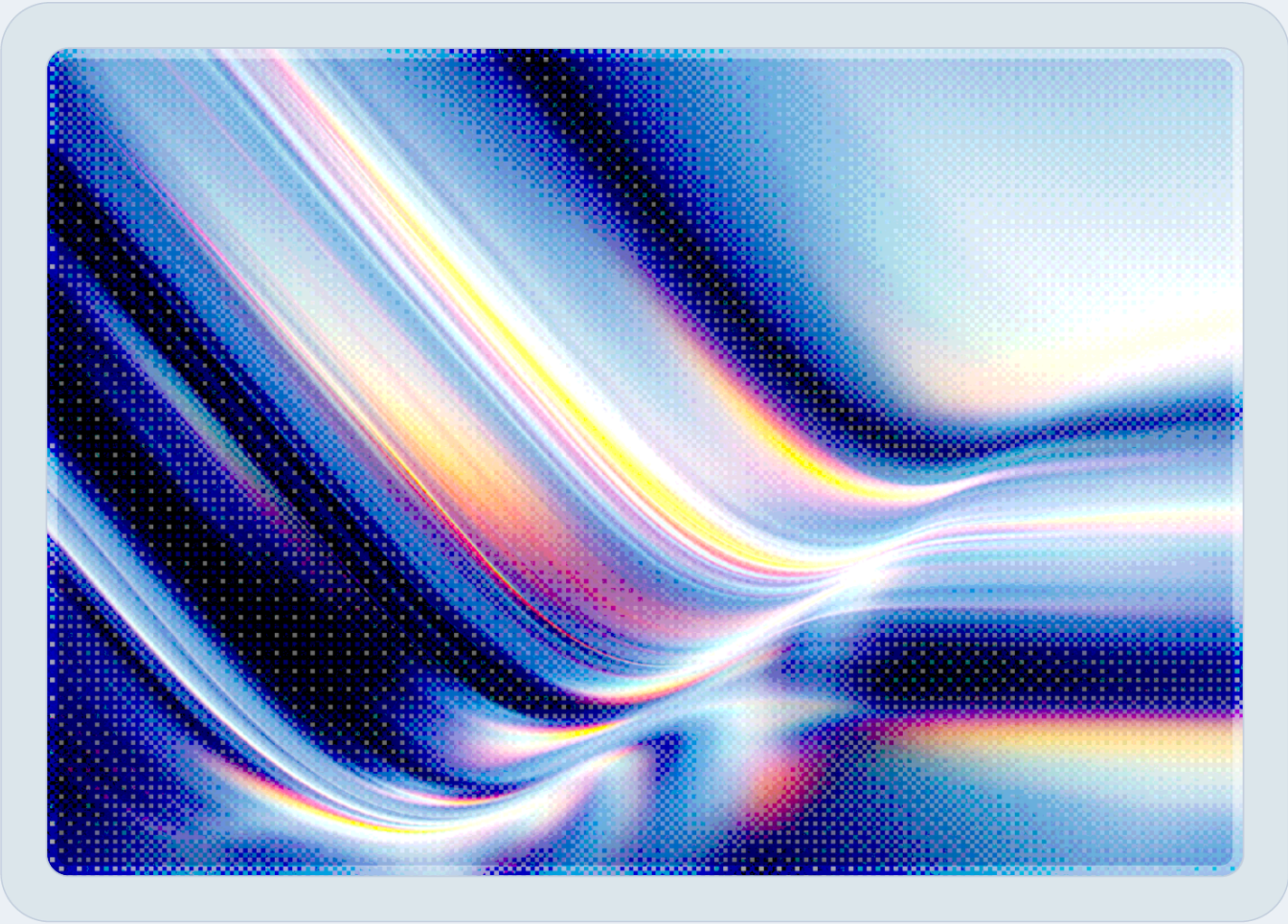
CSAI Foundation | Cloud Security Alliance

Next.js AVIF and Windows Flaws Enable Unauthenticated RCE

2026-08-28

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

Vercel's August 25, 2026 security release for Next.js patched two unrelated, critical-severity vulnerabilities that each permit unauthenticated remote code execution, and the release date was moved up after an additional critical flaw was discovered in an upstream dependency during the disclosure window [1][2]. The first, CVE-2026-75604, is a Windows-specific path traversal flaw (CVSS 3.1: 9.0) affecting any Next.js deployment on a Windows filesystem that uses both the Pages Router and App Router without Cache Components; there is no workaround [3][4], and a working proof-of-concept is already public [11]. The second, tracked as GHSA-2xp9-vwfh-vxw4 (CVSS 4.0: 9.5), traces to a heap buffer overflow in the libheif library that Next.js consumes through the `sharp` image-processing dependency when the Image Optimization API decodes an attacker-supplied AVIF file [2][5]. Both issues are fixed in Next.js 15.5.24 and 16.3.3, and organizations running either affected version range should treat this release as an emergency patch regardless of which vulnerability applies to their deployment [2]. As of this writing, no active exploitation has been confirmed for either flaw [9][10], but the combination of critical severity, no-authentication-required access, and framework ubiquity — Next.js draws roughly 45 million weekly npm downloads [10] — makes rapid, wide-scale scanning and exploitation attempts likely in the days following disclosure.

Background

Next.js is a widely used React application framework, deployed across e-commerce, SaaS, and enterprise web properties for its server-side rendering, routing, and built-in Image Optimization API [10]. That API resizes, re-encodes, and caches images on demand, including images fetched from remote or user-supplied URLs, which makes it a plausible target for attackers, since it accepts external input and performs file and memory operations on the server's behalf. On August 25, 2026, the Next.js core team published its August security release, explaining that the schedule had been accelerated from a previously announced date after engineers identified an additional critical-severity issue in one of the framework's upstream dependencies during the run-up to disclosure [2]. That accelerated issue turned out to be the AVIF decoding flaw described below, and it was bundled into the same release as the Windows path traversal fix that had already been queued.

The two vulnerabilities share a release window but nothing else in terms of root cause, and understanding that distinction matters for triage. CVE-2026-75604 is a platform-specific defect: it exists because Windows and POSIX filesystems interpret path separators and reserved characters differently, and Next.js's routing logic did not fully account for that divergence when applications combined the Pages Router and App Router without Cache Components enabled [3][4]. GHSA-2xp9-vwfh-vxw4, by contrast, is a supply-chain issue: the vulnerable code lives in libheif, an open-source HEIF/AVIF codec library that the `sharp` npm package uses for AVIF decoding, and Next.js inherits the exposure only because its Image Optimization API calls into `sharp` when an application has AVIF output enabled [2][5]. The upstream libheif defect is tracked separately as GHSA-g89c-p67h-r497 in the libheif project itself, underscoring that the vulnerable code was not authored by the Next.js team [6].

Security Analysis

CVE-2026-75604: Windows path traversal to unauthenticated RCE

CVE-2026-75604 carries a CVSS 3.1 base score of 9.0 (vector AV:N/AC:H/PR:N/UI:N/S:C/C:H/I:H/A:H) and is classified under CWE-22, improper limitation of a pathname to a restricted directory [4]. The advisory describes the flaw as affecting Next.js applications that use both the Pages Router and the App Router without Cache Components enabled, when the underlying server filesystem is Windows rather than Linux or macOS [3][4]. Because Windows resolves path separators, drive letters, and certain reserved device names differently than POSIX systems, request-supplied path segments that Next.js's routing and rewrite logic assumed were safely contained within an application directory can, on Windows, resolve outside that directory and ultimately reach code execution [4]. Vercel credits researchers evolutionstorm and B0RI with the responsible disclosure [1][3]. The affected version ranges are Next.js 13.4 through 15.5.23 and 16.0 through 16.3.2; fixed versions are 15.5.24 and 16.3.3 [2][4]. Notably, Vercel's own advisory states plainly that there is no known workaround for affected Windows-hosted applications short of upgrading; in CSA's assessment, that absence of a workaround warrants treating the issue as an emergency patch rather than a routine-cadence item for any organization running Next.js on Windows infrastructure [2]. A public proof-of-concept for CVE-2026-75604 was posted to GitHub the same day as disclosure [11], and infrastructure providers Fastly and Cloudflare both moved quickly to ship virtual-patch WAF rules ahead of customer upgrades [7][8] — signals that this vulnerability is being actively studied by researchers and, plausibly, by opportunistic attackers, even though no confirmed in-the-wild exploitation has been reported as of August 27, 2026 [9][10].

GHSA-2xp9-vwfh-vxw4: AVIF image parsing to unauthenticated RCE

The second vulnerability received a CVSS 4.0 score of 9.5 and stems from how libheif processes maliciously crafted AVIF image containers [5]. According to the upstream libheif advisory, a crafted AVIF file that includes nested identity-derivation and auxiliary item references can cause libheif to construct a decoded image containing two alpha-plane entries at different bit depths, and the resulting mismatch produces out-of-bounds heap writes during image scaling [6]. Because Next.js's Image Optimization API calls `sharp`, which in turn calls libheif to decode AVIF inputs, any Next.js deployment that has AVIF output enabled and processes attacker-influenced image URLs inherits this memory-corruption primitive as a remotely reachable, unauthenticated code-execution path [2][5]. The vulnerable version range is broader than the Windows flaw — Next.js 10.0.0 through versions prior to 15.5.24 and 16.3.3 — but the practical exposure is narrower, since AVIF optimization must be explicitly configured; deployments that have not turned on AVIF output are not exposed regardless of version [1][2]. Vercel's own remediation for this issue in the patched releases is to disable AVIF optimization outright until a corrected libheif build propagates through the dependency chain, rather than to patch the decoding logic directly — a choice consistent with the fact that the defect lives in third-party code Next.js does not control [2]. Vercel credits the Hacktron research team, with researcher rootxharsh identified as finder and KarimPwnz as coordinator; the researchers published a Python proof-of-concept that demonstrates heap corruption under AddressSanitizer [1]. In CSA's assessment, that is a strong technical indicator that a working exploit is achievable without needing memory-layout information specific to a target, though converting that corruption into reliable remote code execution would still require additional engineering.

The table below summarizes both vulnerabilities for triage purposes.

Attribute	CVE-2026-75604 (Windows path traversal)	GHSA-2xp9-vwfh-vxw4 (AVIF decode overflow)
Severity	CVSS 3.1: 9.0 (Critical)	CVSS 4.0: 9.5 (Critical)
Root cause	Path/pathname handling divergence on Windows filesystems (CWE-22)	Heap buffer overflow in upstream libheif AVIF decoder, reached via `sharp`
Trigger condition	Pages Router + App Router without Cache Components, Windows-hosted server	AVIF image optimization explicitly enabled

Attribute	CVE-2026-75604 (Windows path traversal)	GHSA-2xp9-vwfh-vxw4 (AVIF decode overflow)
Affected versions	13.4–15.5.23, 16.0–16.3.2	10.0.0 through <15.5.24 and <16.3.3
Patched versions	15.5.24, 16.3.3	15.5.24, 16.3.3 (AVIF optimization disabled pending upstream fix)
Workaround available	None	Disable AVIF optimization
Public PoC	Yes	Python ASAN proof-of-concept published

Both flaws satisfy the conditions that most concern defenders: no authentication is required, no user interaction is needed, and successful exploitation compromises confidentiality, integrity, and availability of the affected host [4][5]. Neither vendor nor independent researchers had confirmed active exploitation of either issue as of August 27, 2026, but that status can change quickly once technical write-ups and proof-of-concept code circulate publicly, as has already begun for the Windows path traversal issue [7][10].

Recommendations

Immediate Actions

Every organization running Next.js in production should determine, within the next 24 hours, which version is deployed and whether that version falls within the affected ranges for either vulnerability, then upgrade to 15.5.24 (Maintenance LTS) or 16.3.3 (Active LTS) as appropriate [2]. Teams operating Next.js on Windows-hosted infrastructure should treat CVE-2026-75604 as the higher-priority item given the absence of any workaround, and should not rely on network-layer mitigations alone as a substitute for patching [4]. Organizations that cannot patch immediately and that have AVIF optimization enabled should disable it in their Next.js configuration as an interim measure, since that setting is the sole precondition for the libheif-derived exploitation path [2].

Short-Term Mitigations

Where immediate patching is not feasible, teams should apply available virtual patches or WAF rules from their CDN or edge provider — Fastly and Cloudflare both moved quickly to publish rules targeting the Windows path traversal and AVIF decode patterns — while the underlying upgrade is scheduled [7][8]. Security teams should also review logs for anomalous requests to routing endpoints containing traversal sequences or unusual character encodings, and separately review Image Optimization API access logs for AVIF uploads from unfamiliar or unauthenticated sources, since both request classes are consistent with reconnaissance or exploitation attempts against these flaws [9]. Organizations should inventory all Next.js deployments — including those managed by third parties or embedded in vendor products — since the framework's popularity means it is frequently present in systems that are not directly managed by the security team commissioning the review.

Strategic Considerations

This incident is a reminder that a framework's security posture is only as strong as its weakest transitive dependency; the more severe of the two vulnerabilities here originated not in Next.js's own code but in libheif, several layers down the dependency graph through `sharp` [2][5]. Organizations should extend software composition analysis and dependency-vulnerability monitoring beyond direct dependencies to cover the libraries those dependencies pull in, particularly for any component that parses untrusted binary input such as image, document, or media files. The Windows-specific nature of CVE-2026-75604 also illustrates that platform choice is itself a variable in vulnerability exposure, and teams that run identical application code across mixed Windows and Linux fleets should not assume a patch validated on one platform closes exposure on the other. Finally, both flaws reached critical severity specifically because they are reachable without authentication through routine application functionality — routing and image optimization — rather than through an obscure administrative interface, reinforcing that the most consequential vulnerabilities often live in the code paths an application exercises constantly rather than the ones it rarely touches.

CSA Resource Alignment

This incident fits a pattern CSA has previously examined in the Langflow disclosures earlier this year: a widely deployed development or application platform exposes an unauthenticated path-traversal-to-remote-code-execution chain that is exploitable at internet scale before organizations complete patching. CSA's research note on CVE-2026-5027: Langflow Path Traversal to Unauthenticated RCE analyzed the same underlying vulnerability class — improper filename or path sanitization enabling an

unauthenticated actor to write or execute arbitrary code on the server — in a different but architecturally comparable platform, and its recommendations on least-exposure network placement, WAF filtering limitations against encoded traversal sequences, and emergency patch SLAs apply directly to the Next.js findings in this note [12]. CSA's companion analysis, CVE-2026-33017: Langflow RCE Exploits Enterprise AI Pipelines, further documented how quickly opportunistic exploitation can follow disclosure once a working proof-of-concept circulates — a dynamic already underway for CVE-2026-75604 given the public PoC noted above — and reinforces the case for treating no-workaround, no-authentication-required advisories as 24-hour emergency actions rather than routine patch-cycle items [13].

More broadly, both vulnerabilities analyzed here fall within the Threat and Vulnerability Management and Application and Interface Security domains of CSA's AI Controls Matrix (AICM) v1.1, which calls for organizations to maintain dependency inventories, apply emergency patching processes for critical vulnerabilities, and validate that internet-facing services enforce authentication and input validation appropriate to their risk [14]. Although Next.js itself is a general-purpose web framework rather than an AI-specific tool, it is frequently used to build the front ends and API layers of AI-enabled products, and the same AICM controls that govern vulnerability management for AI platforms apply equally to the conventional web infrastructure — such as Next.js-based Image Optimization endpoints — that those AI products depend on.

References

[1] The Hacker News. "Next.js Patches Critical AVIF and Windows Flaws Enabling Unauthenticated RCE." The Hacker News, August 27, 2026.

[2] Next.js. "August 2026 Security Release." Vercel, August 25, 2026.

[3] Vercel/Next.js. "Unauthenticated Remote Code Execution on Windows-hosted servers (GHSA-p293-qw3h-jr36)." GitHub Security Advisories, August 25, 2026.

[4] MITRE/CVE Program. "CVE-2026-75604." CVE Program, August 2026.

[5] Vercel/Next.js. "Unauthenticated Remote Code Execution in Image Optimization API when AVIF files are used (GHSA-2xp9-vwfh-vxw4)." GitHub Security Advisories, August 25, 2026.

[6] libheif Project. "Heap buffer overflow in scale_nearest_neighbor() via duplicate Alpha planes from nested iden/auxl items (GHSA-g89c-p67h-r497)." GitHub Security Advisories, August 2026.

[7] Fastly. "Added virtual patch for CVE-2026-75604 (Next.js Windows Cache Traversal RCE)." Fastly Documentation, August 2026.

[8] Cloudflare. "Emergency WAF release for Next.js CVE-2026-75604 and GHSA-2xp9-vwfh-vxw4." Cloudflare Changelog, August 26, 2026.

[9] Cyber Security News. "Critical Next.js Vulnerabilities Enables Remote Code Execution Attacks." Cyber Security News, August 2026.

[10] Security Online. "45M Weekly Downloads at Risk: Next.js CVE-2026-75604 (CVSS 9.0) Enables Unauthenticated Remote Code Execution." Security Online, August 2026.

[11] rafabd1. "CVE-2026-75604-poc." GitHub, August 25, 2026.

[12] Cloud Security Alliance. "CVE-2026-5027: Langflow Path Traversal to Unauthenticated RCE." Cloud Security Alliance AI Safety Initiative, 2026.

[13] Cloud Security Alliance. "CVE-2026-33017: Langflow RCE Exploits Enterprise AI Pipelines." Cloud Security Alliance AI Safety Initiative, 2026.

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

