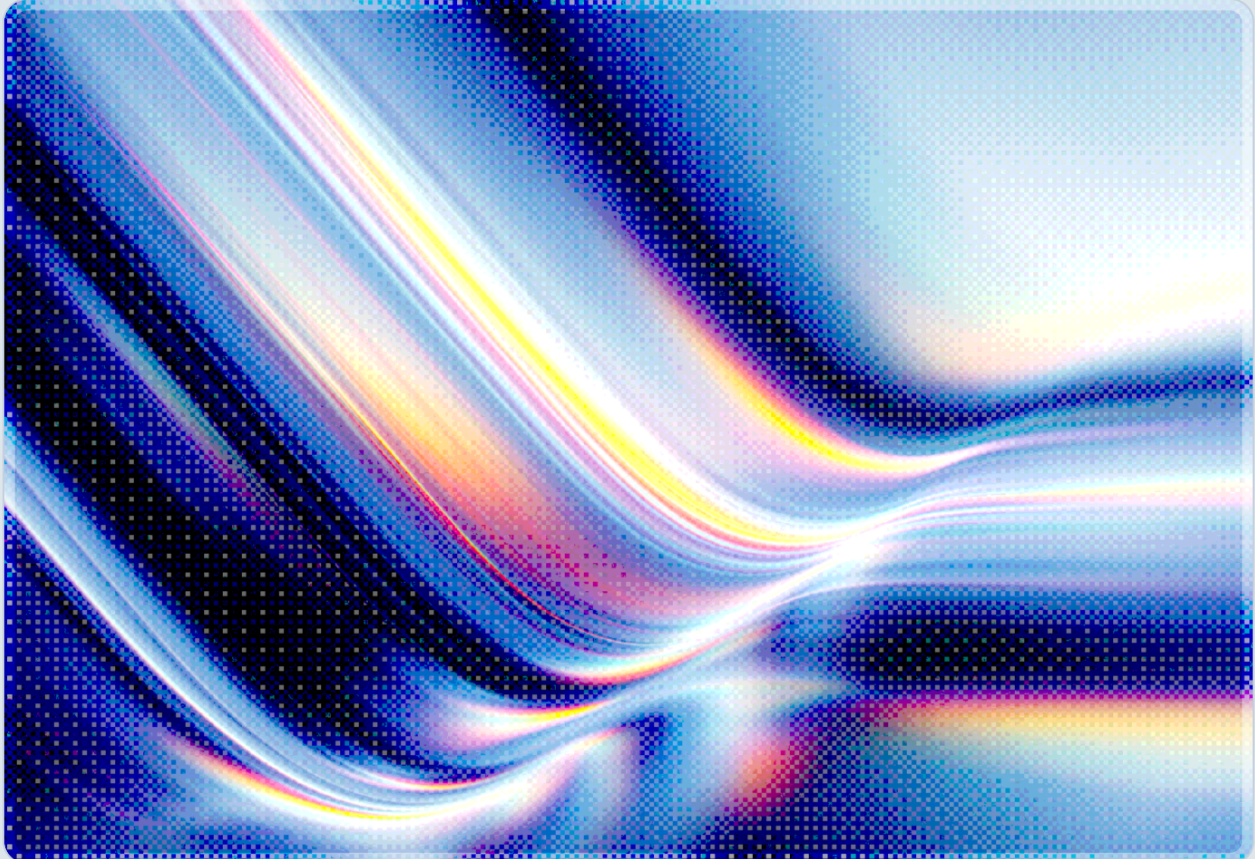


Next.js Out-of-Band Patch: Critical RCE in use cache and libsvg

Security Implications and Guidance

2026-10-16

 AI-assisted Rapid Research



CSAI

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

On October 14, 2026, the Next.js team published an out-of-band security update that fixes two critical, unauthenticated remote code execution (RCE) flaws and one high-severity denial-of-service flaw, in Next.js 16.4.1 and 15.5.28 [1]. Both RCE flaws carry a CVSS score of 9.5 as reported by the Next.js and React advisories [2][3][4]. The first affects App Router applications that use Cache Components and `use cache` functions returning multiple typed arrays (GHSA-673q-mjm6-r5xr). The second is a memory-safety flaw in libsvg, reached through the bundled `sharp` image library, that is exposed through the Image Optimization API (CVE-2026-96889) [1][2].

The `use cache` flaw has no workaround, and Vercel advises upgrading even if an application does not appear to contain a vulnerable function [1]. The SVG flaw has an interim mitigation: ensure `sharp` is version 0.35.5 or later, or block SVG decoding in `sharp` [1][5]. As of October 15, we found no public reports of active exploitation of the three flaws. That status can change quickly once patch diffs are analyzed, so teams should assume the window is short.

Two details matter for scanning and triage. The `use cache` and Server Actions advisories had no CVE identifiers at the time of reporting, so scanners keyed to CVE feeds may not flag them [6]. The SVG flaw is also the third image-processing issue in Next.js dependencies since August [6], which suggests that the image pipeline deserves standing attention as an attack surface.

Background

Next.js is a widely used React framework from Vercel. The App Router with React Server Components (RSC) lets server-side code run as part of request handling, and Cache Components with the `use cache` directive lets developers cache the results of server functions. Next.js also ships an Image Optimization API that resizes and transcodes images on the server, which it does through the native `sharp` library, itself built on libvips and, for SVG, libsvg.

The update is "out-of-band" because Vercel had previously announced, on October 8, that it would publish an unscheduled release on October 14 for three upstream-dependency vulnerabilities (two Critical, one High) [7]. Two of the fixes were postponed from the September 2026 security release because of upstream coordination delays [7][8]. The September release, shipped in 16.3.8 and 15.5.27,

covered a separate set of issues, including image-optimization SSRF (CVE-2026-94483), several cache-poisoning and cache-leak flaws in SSG, ISR and `use cache` handling, and a development-server information disclosure [8]. Teams that patched in September are therefore still exposed to the October flaws unless they moved to 16.4.1 or 15.5.28.

The reliance on upstream components shapes the response. The `use cache` flaw originates in React's RSC server packages, and React published a related advisory, GHSA-xqc2-xf4c-x3h3, for `react-server-dom-webpack`, `react-server-dom-parcel` and `react-server-dom-turbopack`, with fixes in 19.0.9, 19.1.10, 19.2.9 and 19.3.1 [4]. The SVG flaw originates in librsvg, which `sharp` bundles. Applications that use these packages outside of Next.js, for example other RSC frameworks, may need to review the React advisory independently.

Security Analysis

RCE through `use cache` and typed arrays

The Next.js advisory states that an application using the App Router with Cache Components can be exploited for unauthenticated RCE when a `use cache` function returns multiple typed arrays [1]. The React advisory describes crafted payloads against frameworks that support RSC, specifically where responses contain multiple typed arrays alongside attacker-controlled input [4]. Press coverage characterizes the mechanism as memory corruption in typed-array handling and reports that React describes the affected scenario as "extremely rare" [6]. We could not independently confirm the low-level mechanism beyond these summaries, and the advisories do not publish exploit details.

Affected Next.js versions are 16.0.0 through 16.4.0, with the fix in 16.4.1 [3]. The "extremely rare" qualifier describes how few applications are likely to match the vulnerable function pattern. It does not reduce the severity for those that do, and it should not justify delay. The advisory's guidance to upgrade "even if your application does not have similar functions" [1] likely reflects the difficulty of proving a negative: teams may not know which cached functions return typed arrays, for example if one is returned indirectly through serialized nested data. Because no workaround exists, upgrading is the only control that removes the vulnerability.

RCE through librsvg and SVG processing

CVE-2026-96889 is a memory vulnerability in an upstream library used for Image Optimization. The Next.js advisory says it can lead to RCE when Next.js runs on glibc-based Linux [1][2]. Reporting identifies the library as librsvg, accessed through `sharp`, with librsvg 2.63.2 (bundled in `sharp` 0.35.5 and later) containing the fix [5][6]. The `sharp` advisory is rated High (CVSS 4.0 8.9) and associated with CWE-416 (use after free) and CWE-1395 [5]. The Next.js advisory scores the issue higher, at CVSS 4.0 9.5, because it assesses the impact on a network-reachable server [2]. Both ratings use the same scoring system, so the gap reflects different impact assumptions rather than a disagreement about the defect.

Exposure depends on configuration, and sources differ on the details. The Hacker News reports that the flaw is reachable for applications that enable `dangerouslyAllowSVG` or that use `next/og` on the Node.js runtime, and notes that Next.js does not optimize SVGs by default [6]. The GitHub advisory [2] does not list these preconditions as of October 16, and we could not confirm the `next/og` condition against the primary sources. Until Vercel clarifies, defenders should assume any application with the Image Optimization API enabled and a path for untrusted SVG input is potentially exposed, and should not rely on the default setting as a safeguard.

Whether memory corruption becomes code execution depends on the build of the Node.js binary. The `sharp` advisory recommends using a Position Independent Executable (PIE) build of Node.js, which most Linux package managers provide but the official Node.js binaries do not [5]. The `sharp` maintainer cautions that such hardening affects whether an attack results in code execution or a crash [6]. This observation is useful for prioritization but is not a mitigation, since a crash is still a denial of service and hardening does not remove the underlying defect. The flaw was publicly disclosed on September 23 and fixed in `sharp` on September 27, which means the fix existed upstream before the Next.js release [6].

Denial of service in Server Actions

The third flaw (GHSA-p3m5-59mp-jv5r) affects App Router applications with at least one Server Action. Crafted requests can drive CPU usage high enough to block other requests in the same process [1]. Pages Router applications and those without Server Actions are not affected, and no workaround is available [1][11]. The related React advisory is GHSA-h3gv-c5rc-p453 [1].

Summary of the October 14 fixes

Issue	Identifier	Severity	Affected Next.js	Fixed in	Workaround
RCE, <code>use cache</code> typed arrays	GHSA-673q-mjm6-r5xr	Critical (CVSS 9.5)	16.0.0–16.4.0	16.4.1	None
RCE, SVG via <code>librsvg/ sharp</code>	CVE-2026-96889 / GHSA-2464-fhr4-w9xw	Critical (CVSS 9.5)	10.0.0–15.5.27, 16.0.0–16.4.0	15.5.28, 16.4.1	<code>sharp ≥ 0.35.5</code> , or block SVG decoding
DoS, Server Actions	GHSA-p3m5-59mp-jv5r	High	13.0.0–15.5.27, 16.0.0–16.4.0	15.5.28, 16.4.1	None

Sources: [1][2][3][11]. Version ranges were confirmed against the GitHub advisories.

Exploitation status and detection gaps

At the time of writing, we found no public reports of in-the-wild exploitation. Two factors may nonetheless shorten the time to exploitation. First, the SVG flaw's fix and disclosure have been public since late September, so attackers have had weeks in which they could have studied the upstream change. Second, the `use cache` and Server Actions advisories lacked CVE identifiers and did not appear in npm's advisory feed as of October 15, so `npm audit` and software composition analysis tools may not report them [6]. Organizations that rely only on CVE-based alerts could therefore miss the two issues that have no workaround. This gap is likely to close as identifiers are assigned, but teams should not wait for it.

The broader pattern is also worth noting. The Hacker News counts three image-processing vulnerabilities in Next.js dependencies since August: a libheif AVIF flaw, a `next/og` SVG generation flaw in September, and this `librsvg` flaw [6]. This is consistent with a recurring pattern of memory-safety flaws in native media parsers. It is also consistent with CSA's analysis that machine-speed vulnerability discovery is outpacing the ability of open-source maintainers and downstream consumers to patch [9].

Recommendations

Immediate Actions

Upgrade Next.js to 16.4.1 on the 16.x line or 15.5.28 on the 15.x line, then rebuild and redeploy, since the fixed `sharp` version is delivered through the build [1]. After the build, confirm the resolved versions with your package manager (`npm ls sharp` , and the lockfile for React server packages) rather than assuming the upgrade propagated. This release states no fix for applications on versions older than 15.5, so they need to move to a supported line. If an application uses RSC packages directly, upgrade them to the versions listed in the React advisory [4].

For teams that cannot deploy immediately, apply the SVG mitigations. Pin `sharp` to 0.35.5 or later, or block SVG decoding with `sharp.block({ operation: ["VipsForeignLoadSvg"] })` [5]. Disable `dangerouslyAllowSVG` where it is not needed. Since the `use cache` and Server Actions flaws have no workaround, prioritize internet-facing applications that use Cache Components or Server Actions, and consider temporary controls at a web application firewall or reverse proxy for rate limiting, understanding these do not remove the RCE risk.

Short-Term Mitigations

Run the Node.js process with least privilege, and review whether a successful RCE in the web tier could reach cloud credentials, build secrets or internal networks. Prefer a PIE build of Node.js on Linux hosts and containers, as the `sharp` maintainers advise [5], while recognizing it only reduces exploit reliability. Where image processing handles untrusted uploads, consider isolating it in a separate service or sandbox with no access to application secrets.

Add detection that does not depend on CVE feeds. Track the Next.js and React GitHub security advisories directly, and subscribe to the Vercel security announcements, which gave a week of advance notice in this case [7]. Review application and image-optimization logs from at least October 14 backward to September 23 for anomalous SVG requests, unexpected process crashes, and spikes in CPU usage on App Router servers.

Strategic Considerations

This release shows how a single upstream dependency delay can leave a framework's users exposed to a known issue for weeks. Teams should keep an inventory of transitive native dependencies in their web tier, including image and media libraries, and should be prepared to ship emergency updates outside their normal change windows. Treating framework patch releases as routine, rather than as a standing capability to ship within days, is likely to increase risk, as the interval between the upstream fix and the Next.js release in this case illustrates. Organizations might also evaluate whether some of the image-processing work can be moved to memory-safe components or to managed services.

CSA Resource Alignment

CSA's [Project Glasswing: AI Discovery Outpaces Open Source Patching Capacity](#) [9] is the most directly relevant prior work. It describes the widening gap between rapid vulnerability discovery and the slower pace at which open-source maintainers and downstream users patch. The Next.js sequence, in which fixes for two critical flaws were delayed by upstream coordination and the SVG fix reached `sharp` weeks before it reached Next.js applications, illustrates that gap in a widely deployed framework. The paper's guidance on shortening patch cycles and tracking upstream dependencies applies directly.

The [AI Controls Matrix \(AICM\) v1.1](#) [10] supplies the control vocabulary for the vulnerability-management response. Its Threat and Vulnerability Management (TVM) domain covers patching and remediation timelines, and its Application and Interface Security (AIS) domain covers secure handling of untrusted input in application components. These are relevant to AI-enabled web applications that are built on frameworks such as Next.js, which is frequently used to front AI applications and agents. This note does not assert that the flaws are specific to AI workloads.

References

- [1] Vercel / Next.js. "[Next.js Security Update for Upstream Vulnerabilities](#)." Next.js Blog, October 14, 2026.
- [2] Vercel. "[Next.js Image Optimization API RCE \(CVE-2026-96889, GHSA-2464-fhr4-w9xw\)](#)." GitHub Security Advisories, October 14, 2026.
- [3] Vercel. "[Unauthenticated Remote Code Execution in use cache functions returning typed arrays \(GHSA-673q-mjm6-r5xr\)](#)." GitHub Security Advisories, October 14, 2026.
- [4] React. "[Unauthenticated Remote Code Execution in React Server Components \(GHSA-xqc2-xf4c-x3h3\)](#)." GitHub Security Advisories, October 2026.
- [5] sharp. "[GHSA-wq5f-xc86-pv6w: sharp / libsvg memory corruption](#)." OSV, September 2026.
- [6] The Hacker News. "[Next.js Ships Out-of-Band Patch for Two Critical RCE Flaws in Caching and SVG Processing](#)." The Hacker News, October 2026.
- [7] Vercel / Next.js. "[Upcoming Next.js Security Update for Upstream Vulnerabilities](#)." Next.js Blog, October 8, 2026.
- [8] Vercel / Next.js. "[September 2026 Security Release](#)." Next.js Blog, September 30, 2026.
- [9] Cloud Security Alliance. "[Project Glasswing: AI Discovery Outpaces Open Source Patching Capacity](#)." CSA, June 7, 2026.
- [10] Cloud Security Alliance. "[AI Controls Matrix v1.1](#)." CSA, June 22, 2026.
- [11] Vercel. "[Denial of Service in Server Actions \(GHSA-p3m5-59mp-jv5r\)](#)." GitHub Security Advisories, October 14, 2026.