ultraviolet

THREAT ADVISORY

# GitHub Enterprise Vulnerability

**Services Performed By:**
UltraViolet Cyber TIDE Team
tide@uvcyber.com

**Published Date:**
April 29, 2026
TLP:GREEN

# Executive Snapshot

CVE-2026-3854 is a critical command injection vulnerability in GitHub's internal git infrastructure, publicly disclosed by Wiz Research on April 28, 2026, that allows any authenticated user with repository push access to achieve remote code execution on backend servers — including cross-tenant access to millions of repositories on GitHub.com — using nothing more than a single git push command. The flaw exploits unsanitized semicolons in git push option values that are embedded into an internal X-Stat header, enabling an attacker to inject metadata fields that override security-critical configuration, bypass sandboxing, and execute arbitrary commands as the git service user. Notably, this is one of the first critical vulnerabilities in closed-source binaries discovered using AI-augmented reverse engineering, specifically IDA MCP, signaling a shift in how complex multi-binary systems will be audited by both researchers and adversaries going forward. Organizations should take the following steps immediately:

- Upgrade all GitHub Enterprise Server instances to a patched version (3.14.24, 3.15.19, 3.16.15, 3.17.12, 3.18.6, or 3.19.3) without delay, as 88% of GHES instances were still vulnerable at the time of public disclosure and the exploitation barrier is trivially low.
- Audit all repository push access permissions, service accounts, and automation tokens to enforce least-privilege principles, ensuring that only explicitly authorized users and workflows retain write access.
- Review git push option usage across CI/CD pipelines and developer tooling for anomalous or unexpected values and conduct a proactive threat hunt against git operation logs covering the window between March 4, 2026, and the date your instance was patched.
- Initiate an architecture review of any internal service-to-service protocols that pass user-controlled input through shared data formats, treating delimiter-based interchange headers with the same input validation rigor applied to external-facing APIs.

# TIDE Team Analysis

On April 28, 2026, Wiz Research publicly disclosed CVE-2026-3854, a critical command injection vulnerability in GitHub's internal git infrastructure affecting both GitHub.com and all supported versions of GitHub Enterprise Server. Carrying a CVSS score of 8.7, the flaw allows any authenticated user with push access to a repository to execute arbitrary commands on GitHub's backend servers using nothing more than a standard git client and a single git push command. The vulnerability was reported to GitHub on March 4, 2026, and GitHub deployed a fix to GitHub.com the same day. However, at the time of public disclosure, 88% of GitHub Enterprise Server instances remained unpatched, making immediate remediation an urgent priority for any organization running self-hosted GHES.

The root cause is a delimiter injection flaw in GitHub's internal protocol. When a user runs git push, the request flows through several internal components, including babeld (a git proxy), gitauth (an authentication service), and gitrpcd (an internal RPC server), all of which communicate via an internal X-Stat header containing semicolon-delimited key=value pairs. The critical weakness is that babeld copies git push option values directly into the X-Stat header without sanitizing semicolons. Because the semicolon is also the field delimiter, an attacker can break out of the push option field and inject arbitrary metadata entries. Compounding this, the header parser uses last-write-wins semantics, meaning injected fields that appear later in the header silently override their legitimate counterparts, including security-critical configuration values.

The exploitation chain escalates from header injection to full remote code execution through three discrete steps. First, the attacker injects a non-production rails_env value to switch from a sandboxed execution path to an unsandboxed one that runs hooks directly as the git service user. Second, a custom_hooks_dir injection redirects the base directory where the binary looks up hook scripts. Third, a crafted repo_pre_receive_hooks entry containing a path traversal sequence causes the binary to resolve and execute an arbitrary binary on the filesystem. Each injection alone is insufficient, but chained together they completely bypass GitHub's sandboxing protections and yield code execution as the git user.

This vulnerability was not limited to GitHub Enterprise Server. When Wiz initially ran the exploitation chain against GitHub.com, custom hooks never executed because a boolean enterprise mode flag defaults to false on GitHub.com. However, since that flag is also carried in the X-Stat header, it was equally injectable through the same mechanism. With one additional injected field, the full chain worked on GitHub.com, yielding code execution on production infrastructure. The cross-tenant implications are severe. GitHub.com is a multi-tenant platform where repositories belonging to millions of different organizations and users are stored on shared backend infrastructure, and the git user has broad filesystem access to every repository hosted on the compromised node. Wiz confirmed that millions of repository index entries across multiple organizations were accessible from compromised nodes, though they validated this exposure only against their own test accounts.

Organizations running GitHub Enterprise Server should treat patching as an emergency action. Beyond patching, security teams should audit git push option usage across CI/CD pipelines and developer workflows, review push access permissions on a least-privilege basis, and inventory all service accounts or automation tokens with write access to repositories. Organizations with high-value repositories should consider a proactive threat hunt focused on git operation logs covering the window between March 4, 2026, and the date their instance was patched, though GitHub has stated there is no evidence of in-the-wild exploitation.

This vulnerability illustrates a systemic architectural risk that extends well beyond GitHub. When multiple services written in different languages pass data through a shared internal protocol, the assumptions each service makes about that data become a critical attack surface. In this case, babeld assumed push option values were safe to embed verbatim. gitrpcd assumed every field in the X-Stat header was set by a trusted source. The pre-receive hook

assumed an environment variable could only be "production" in production. Each assumption was reasonable in isolation and dangerous in combination. CTOs and CISOs operating microservices or multi-service architectures should audit how user-controlled input flows through internal protocols, particularly wherever security-critical configuration is derived from shared data formats.

The broader industry significance of this finding lies in the methodology behind its discovery. Wiz explicitly notes this is one of the first critical vulnerabilities discovered in closed-source binaries using AI. By leveraging AI-augmented reverse engineering tooling, particularly IDA MCP, the researchers rapidly analyzed GitHub's compiled binaries, reconstructed internal protocols, and systematically identified where user input could influence server behavior across the entire pipeline. Previous audit attempts against the same codebase were impractical due to the sheer volume of compiled blackbox binaries involved, but IDA MCP made the analysis feasible at a speed that would not have been achievable manually. This represents a meaningful inflection point, as AI-assisted tooling is now enabling researchers to find deep, cross-component vulnerability classes in closed-source systems that historically went unexamined.

For security leadership, the actionable takeaways are clear. Patch GHES immediately, as the fixed versions are 3.14.24, 3.15.19, 3.16.15, 3.17.12, 3.18.6, and 3.19.3, and the exploitation bar is trivially low for any authenticated user with push access. Internalize that internal service protocols are not trust boundaries, and treat delimiter-based data interchange formats with the same rigor as external API inputs. Recognize that the offensive research landscape is shifting. AI-augmented reverse engineering tools like IDA MCP are compressing the time required to audit complex, multi-binary systems, and threat actors will adopt these same capabilities. Organizations should be investing in equivalent defensive tooling and prioritizing architecture reviews that map user-controlled data flows across internal service boundaries before the next critical flaw is found by someone less responsible.

# Why It Matters

The organizational risk posed by CVE-2026-3854 cannot be overstated. GitHub Enterprise Server is deeply embedded in the software development lifecycle of most modern enterprises, serving as the central repository for proprietary source code, infrastructure-as-code definitions, deployment secrets, and CI/CD pipeline configurations. A vulnerability that grants an authenticated user full filesystem access to the underlying server, with no specialized tooling, no privilege escalation chain, and no social engineering required, effectively turns every developer with push access into a potential vector for complete infrastructure compromise. For organizations subject to regulatory frameworks, the cross-tenant exposure demonstrated on GitHub.com raises additional concerns around data residency, intellectual property protection, and third-party risk, particularly for firms that rely on GitHub's shared infrastructure to host sensitive codebases alongside those of other tenants.

What makes this disclosure particularly consequential for security leadership is not just the vulnerability itself, but how it was found. Wiz Research was transparent that previous attempts to audit the same GitHub binary pipeline were abandoned because the volume of closed-source, compiled components made manual reverse engineering impractical. The introduction of IDA MCP, an AI-augmented reverse engineering capability integrated into the industry-standard IDA disassembler, fundamentally changed that calculus. By using AI to rapidly reconstruct internal protocols, trace data flows across service boundaries, and identify injection points in compiled binaries, the Wiz team compressed what would have been months of manual analysis into a timeframe that made the research viable. This is not an incremental improvement in tooling; it represents a structural shift in what is discoverable within complex, closed-source systems.

The implications for organizational security strategy are twofold. On the defensive side, security teams should recognize that the attack surface of closed-source vendor software, previously shielded by the practical difficulty of binary analysis, is now far more accessible to skilled researchers and, inevitably, to adversaries. Vendor assurances that compiled code is opaque enough to resist scrutiny no longer hold. On the offensive and proactive side, organizations with mature security programs should evaluate AI-augmented reverse engineering tools like IDA MCP as part of their own vendor risk assessment and red team capabilities, using them to audit the third-party binaries running in their environments before someone else does. The era in which compilation served as a de facto security boundary is closing, and leadership teams that fail to adjust their threat models accordingly will find themselves reacting to disclosures rather than anticipating them.

# How to Respond

- Strictly adhere to cybersecurity Fundamentals and ensure all personnel undergo annual phishing and social engineering training. Speak with your UltraViolet Cyber TAM Representative to schedule a live phishing engagement.
- Patch all organizationally hosted GitHub Enterprise instances to 3.14.24, 3.15.19, 3.16.15, 3.17.12, 3.18.6, 3.19.3, or later.
- Perform annual tech refresh reviews to gain a holistic understanding of your infrastructure. Speak with your UltraViolet Cyber TAM Representative to schedule a red team or purple team engagement to gain insight into the vulnerabilities in your environment.

# What UltraViolet Cyber is Doing

- Exploring reverse engineering and vulnerability analysis using LLM MCP tools.
- Proactively enabling custom detections based on the collected artifacts, tactics, techniques, and procedures identified in this activity.
- Performing hypothesis driven threat hunts based on threat actor behavior and artifacts. UVCyber customers will be informed of the results through secure channels.
- Parsing available victim dump data for any social, financial, business, or technical relations to UVCyber Clients and partner organizations.
- Aggregating threat intelligence from myriad sources and applying the most up-to-date knowledge to proactive threat hunting and response.

---

## About UltraViolet Cyber

UltraViolet Cyber is a leading tech-enabled managed security services provider, delivering unparalleled cybersecurity expertise that fills technology and talent gaps across Global 2000 and Federal Government customers. Founded and operated by security practitioners from the national intelligence community, UltraViolet Cyber connects offensive security, application security, detection and response, and security engineering to deliver a differentiated approach to cybersecurity operations. Transforming customers' security programs, UltraViolet Cyber's flagship security-as-a-service solution, UV Lens, removes complex operational silos, replacing them with integrated security capabilities. UltraViolet is headquartered in McLean, Virginia with technology centers across the world.

443.351.7630 / info@uvcyber.com | UltraViolet Cyber | @uv_cyber

---