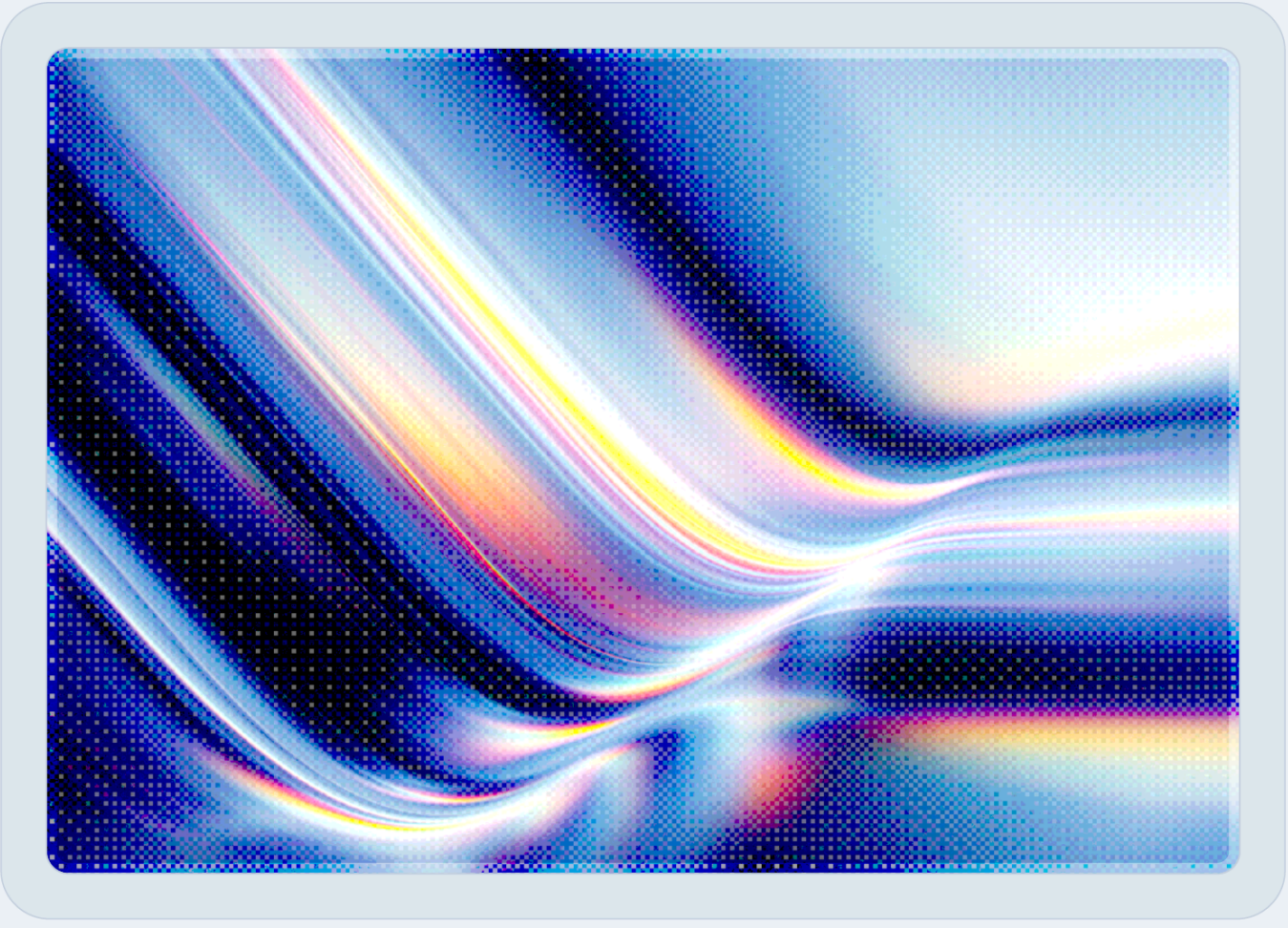
CSAI Foundation | Cloud Security Alliance

Ray Framework RCE Under Active Exploitation Joins CISA KEV

CVE-2025-62593 Exposes AI/ML Compute Clusters via DNS Rebinding

2026-08-18

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

CVE-2025-62593 is a critical (CVSS 4.0 base score 9.4) remote code execution vulnerability in Ray, the open-source distributed computing framework that underlies a large share of production AI and ML training and serving infrastructure, and it affects every version prior to 2.52.0 [1][2]. The flaw stems from Ray's reliance on checking whether an incoming HTTP request's User-Agent header begins with the string "Mozilla" as its sole defense against browser-originated calls to the Ray dashboard's job-submission API, a check that the Fetch API specification — as implemented in Firefox and Safari — permits client-side JavaScript to bypass entirely [2][3]. Combined with a DNS rebinding attack, this weakness lets a malicious website or advertisement redirect a victim's Firefox or Safari browser to a locally or internally reachable Ray instance and submit arbitrary shell commands through the `/api/jobs` and `/api/job_agent/jobs/` endpoints, all without the victim clicking anything beyond loading the page [2][3]. CISA added CVE-2025-62593 to its Known Exploited Vulnerabilities (KEV) catalog on August 17, 2026, confirming exploitation in the wild and giving Federal Civilian Executive Branch agencies until August 20 to remediate [1][4][5][7]. Independent research has already tied the flaw to the RondoDox DDoS botnet, which incorporated a proof-of-concept exploit into its toolkit two days before the vulnerability's public disclosure on November 26, 2025 [6]. Ray 2.52.0 fixes the underlying defect and adds an optional, disabled-by-default token authentication feature; organizations running Ray anywhere in their development, CI/CD, or production AI compute pipeline should treat upgrading as an urgent priority [2][3].

Background

Ray is an open-source Python framework, originally developed at UC Berkeley's RISELab and now stewarded commercially by Anyscale, for scaling machine learning and general-purpose Python workloads across clusters of machines and GPUs. It has become a foundational orchestration layer for distributed model training, hyperparameter tuning, reinforcement learning, and increasingly for serving large language model inference at scale, making it a common dependency across developer workstations, continuous integration runners, container images, and cloud-hosted GPU clusters. Its GitHub repository carried more than 43,500 stars and 7,900 forks as of this writing, a rough indicator of how widely it has been adopted across the ML engineering community [12]. Ray ships with a web-based dashboard, typically listening on port 8265, that exposes a job-submission API intended to let

developers and automated tooling launch and monitor distributed jobs without additional friction. That convenience-first design, consistent with Ray's stated assumption that it runs inside a trusted network boundary, helps explain why the framework has now drawn two consecutive authentication-related vulnerabilities rather than a single isolated incident.

This is not the first time an authentication gap in Ray's dashboard has drawn active exploitation. In 2023, researchers disclosed CVE-2023-48022, a missing-authentication flaw in the same job-submission API that Ray's maintainers characterized as consistent with the framework's intended trust model rather than a bug requiring a patch, and left unresolved [8]. That unresolved gap remained exploitable for two years: in November 2025, security researchers at Oligo Security documented "ShadowRay 2.0," a self-propagating botnet campaign that used CVE-2023-48022 to compromise more than 230,000 internet-exposed Ray servers, many running NVIDIA A100 GPUs, for cryptocurrency mining, data and credential theft, and denial-of-service operations [8][9]. CVE-2025-62593 is a distinct vulnerability, discovered independently by researchers at Socket and Oligo Security and disclosed through GitHub Security Advisory GHSA-q279-jhrf-cc6v on November 26, 2025, but it reflects the same underlying pattern of Ray treating network-level trust as a substitute for authentication, this time in a defense mechanism that was specifically intended to close off browser-based abuse of that trust assumption [2][3].

Security Analysis

The technical root cause of CVE-2025-62593 is narrow but consequential. Ray's dashboard rejects HTTP requests to sensitive endpoints when the User-Agent header starts with "Mozilla," on the theory that legitimate automated clients, such as the Ray CLI or CI/CD tooling, would never send a browser-style header, while a request originating from an actual browser would. The flaw is that the Fetch API specification, implemented in Firefox and Safari, allows client-side JavaScript to set an arbitrary User-Agent value on outgoing requests, so a script running on an attacker-controlled page can simply omit or alter the "Mozilla" prefix and defeat the check entirely [2][3]. An attacker pairs this with a DNS rebinding technique: a victim visits a malicious site or is served a malicious advertisement, and the page's JavaScript first resolves the attacker's domain to an external IP address to pass the browser's initial same-origin checks, then rapidly rebinds that same domain name to the victim's local network address, typically 127.0.0.1 or an internal cluster address where Ray's dashboard is listening on port 8265 [3]. Because the browser continues to treat all subsequent requests as same-origin, the script can then send a POST request carrying a spoofed User-Agent and an arbitrary shell command as the job payload, and Ray will execute it with the privileges of the process running the dashboard [2][3]. Proof-of-concept demonstrations of the flaw have shown OS-agnostic command execution across Windows, macOS, and Linux hosts, requiring only that the victim's browser load the malicious page [2][3].

This mechanism carries two distinct implications for risk exposure that organizations should evaluate separately. The first concerns Ray instances that are deliberately kept off the public internet, for example a developer's local workstation running `ray start --head` for iterative testing, or a cluster dashboard reachable only from an internal corporate network. These instances are exposed specifically through the browser-and-DNS-rebinding chain described above: a developer does not need to misconfigure anything, only to browse the web from a machine or network segment where a Ray dashboard happens to be reachable [2][3]. The second concerns Ray dashboards that are already reachable from the public internet, whether through misconfiguration, an intentionally permissive cloud security group, or a container deployment that binds the dashboard port to all interfaces. For these instances, the User-Agent check that CVE-2025-62593 defeats is not a meaningful barrier to begin with: any direct HTTP client, including the default configurations of common tools such as curl or Python's requests library, sends a User-Agent that does not start with "Mozilla," and would satisfy Ray's flawed check without needing any DNS rebinding step at all.

This reading is consistent with public reporting that automated scanning tools, rather than only browser-mediated social engineering, have incorporated this vulnerability; researchers at BitSight documented that the RondoDox DDoS botnet added an exploit for this flaw to its arsenal two days before the CVE's public disclosure, evidently after monitoring security research and proof-of-concept repositories directly rather than waiting on formal advisories [6]. Notably, the specific exploit code RondoDox deployed set its User-Agent string to "Mozilla/5.0 (rondo2012@atomicmail[.]io)," a value that itself begins with "Mozilla" and would trigger Ray's rejection logic, suggesting an implementation error in the botnet's tooling rather than any resilience in Ray's defense [6]. That detail should not be read as reassurance: it indicates only that one specific botnet's exploit was flawed, not that internet-exposed Ray dashboards are otherwise protected.

Organizations should also distinguish this incident clearly from the 2023 ShadowRay and 2025 ShadowRay 2.0 campaigns, which exploited CVE-2023-48022, a separate and still-unresolved missing-authentication issue in the same job-submission API [8][9]. CVE-2025-62593 does not require Ray to be internet-exposed to be exploited, and organizations that have already restricted Ray dashboards to private networks in response to ShadowRay reporting should understand that this newer flaw can still reach those same restricted instances through a victim's browser. Conversely, organizations that have deployed Ray only in fully air-gapped or highly restricted environments, with no browsing access from any host on the same network segment as the Ray dashboard, face a substantially narrower exposure window for this specific flaw, though the underlying absence of authentication remains a latent risk that any future misconfiguration could expose.

Recommendations

Immediate Actions

Organizations should inventory every Ray deployment across developer workstations, CI/CD runners, container images, Kubernetes workloads, and cloud-hosted data-processing or model-training clusters, then upgrade all instances to Ray 2.52.0 or later without delay [1][2]. Because Ray is frequently installed as a transitive dependency of other ML tooling rather than deployed as a standalone service, teams should scan software bills of materials and container base images for bundled Ray installations that might otherwise be missed by infrastructure inventories focused only on directly provisioned services. Any Ray dashboard reachable from the public internet should be treated as an emergency regardless of patch timeline, and should be moved behind network controls that block direct external access while the upgrade is completed.

Short-Term Mitigations

Beyond patching, organizations should enable the token-based authentication feature that Ray 2.52.0 introduces, since it ships disabled by default and therefore requires an explicit configuration change to take effect [2][3]. Security teams should restrict Ray dashboard access to a minimal set of trusted hosts using firewall rules or network policies rather than relying on any header-based or application-layer check as a substitute for network segmentation, and should specifically ensure that developer workstations and CI/CD runners with Ray installed are not on network segments that also permit unrestricted web browsing. Given that RondoDox and similar automated tooling have already incorporated exploit code for this flaw, organizations should review Ray dashboard access logs for POST requests to `/api/jobs` or `/api/job_agent/jobs/` originating from unexpected source addresses or carrying anomalous User-Agent strings, and treat any Ray instance that has been internet-reachable since before the November 2025 disclosure as a candidate for compromise investigation.

Strategic Considerations

The recurrence of authentication-related vulnerabilities in Ray's dashboard, spanning CVE-2023-48022 in 2023 and CVE-2025-62593 in 2025, reflects a structural pattern in which distributed computing and orchestration frameworks built for trusted-network operation are increasingly deployed in environments where that trust boundary cannot be reliably guaranteed, whether because of developer browsing habits, cloud misconfiguration, or multi-tenant infrastructure. Organizations building AI platforms on Ray or similar frameworks should treat authentication and network isolation for orchestration dashboards as a

standing architectural requirement rather than a one-time remediation, and should include these components in the same vulnerability management and asset discovery programs applied to production-facing services rather than exempting development tooling from routine scrutiny. Because Ray sits at the compute and orchestration layer of many AI pipelines, a compromised instance can expose not only the host itself but also any GPU resources, data, and credentials accessible to the jobs it runs, making its security posture a proportionately higher priority than the framework's "internal tool" origins might suggest.

CSA Resource Alignment

CSA's "Legacy Infrastructure: The AI Agent Security Blind Spot" is the most directly applicable prior CSA work, since it names Ray explicitly among the orchestration and compute frameworks that constitute the "pre-AI vulnerabilities with post-AI blast radius" pattern this incident continues, and it already documents the 230,000-plus internet-exposed Ray servers implicated in the ShadowRay 2.0 campaign as a benchmark for the scale of exposure this class of framework can carry [10]. That research note's recommendation to build a substrate inventory of every Ray cluster and verify authentication status applies directly to CVE-2025-62593, with the added nuance that this newer flaw defeats even instances that organizations believed they had adequately isolated from direct internet exposure. CSA's "LiteLLM AI Gateway: Active Exploitation via MCP Injection" offers a structurally similar precedent: another widely deployed piece of AI infrastructure tooling, built with convenience-first defaults, reached CISA KEV status through a chain of authentication and validation weaknesses, and that note's emphasis on treating AI middleware and orchestration layers as security-critical rather than incidental supports the same conclusion here.

More broadly, CVE-2025-62593 illustrates a gap in the identity and access management and threat and vulnerability management domains of CSA's AI Controls Matrix (AICM) v1.1: a component central to AI compute orchestration relied on an easily defeated header check in place of genuine authentication, and shipped its actual authentication feature disabled by default [11]. Organizations assessing Ray or comparable distributed computing frameworks against AICM's identity and access management domain should verify that authentication is explicitly enabled rather than assumed, and should extend threat and vulnerability management processes to cover orchestration and compute-cluster dashboards with the same rigor applied to externally facing production services.

References

[1] The Hacker News. "CISA Flags Actively Exploited Ray Flaw That Can Trigger Browser-Based RCE." The Hacker News, August 2026.

[2] Ray Project. "Critical RCE Vulnerability against Ray Devs exploitable via Browser (Safari & Firefox) due to DNS Rebinding Attack — GHSA-q279-jhrf-cc6v." GitHub Security Advisories, November 26, 2025.

[3] National Institute of Standards and Technology. "CVE-2025-62593 Detail." National Vulnerability Database, November 26, 2025.

[4] Cybersecurity and Infrastructure Security Agency. "CISA Adds One Known Exploited Vulnerability to Catalog." CISA, August 17, 2026.

[5] The Register. "CISA Gives Feds 3 Days to Fix Actively Exploited Ray RCE Bug." The Register, August 18, 2026.

[6] BitSight. "RondoDox Botnet: From Zero to 174 Exploited Vulnerabilities." BitSight Blog, March 11, 2026.

[7] Security Affairs. "U.S. CISA Adds a Ray-Project Ray Flaw to Its Known Exploited Vulnerabilities Catalog." Security Affairs, August 2026.

[8] Oligo Security. "ShadowRay 2.0: Attackers Turn AI Against Itself in Global Campaign That Hijacks AI into Self-Propagating Botnet." Oligo Security Blog, November 2025.

[9] The Hacker News. "ShadowRay 2.0 Exploits Unpatched Ray Flaw to Build Self-Spreading GPU Cryptomining Botnet." The Hacker News, November 2025.

[10] Cloud Security Alliance. "Legacy Infrastructure: The AI Agent Security Blind Spot." Cloud Security Alliance, June 2026.

[11] Cloud Security Alliance. "AI Controls Matrix (AICM) v1.1." Cloud Security Alliance, 2025.

[12] Ray Project. "ray-project/ray." GitHub repository, accessed August 18, 2026.

