CSAI Foundation

Cloud Security Alliance AI Safety Initiative

Marimo Pre-Auth RCE Weaponized for Blockchain Botnet

CVE-2026-39987 Exploited to Deploy NKAbuse Variant via HuggingFace Spaces

Unofficial AI-assisted Research

2026-04-20

Key Takeaways

A critical unauthenticated remote code execution vulnerability in the Marimo Python notebook platform — CVE-2026-39987, CVSS 9.3 [11] — was disclosed on April 8, 2026, and weaponized within three days by threat actors deploying a blockchain-controlled botnet payload hosted on HuggingFace Spaces. The root cause is a missing authentication check on the `/terminal/ws` WebSocket endpoint, which grants any unauthenticated network client a full interactive shell on the host system. From April 11 to April 14, researchers at Sysdig observed 662 exploit events originating from 11 unique IP addresses across 10 countries [2][10]. The most significant campaign delivered a new variant of NKAbuse — a Go-based backdoor that uses the NKN decentralized blockchain network for command-and-control (C2) — from a typosquatted HuggingFace Space named "vsccode-modetx" [2].

The incident has immediate implications beyond patching. It demonstrates that AI development tooling can face the same rapid weaponization curve as enterprise software — and that at least some threat actors are already treating it that way. It also establishes that trusted AI platform infrastructure — specifically HuggingFace Spaces — is being exploited as a malware delivery channel, allowing attackers to bypass URL filtering and domain reputation controls by abusing a recognized, widely-allowlisted hostname. Organizations should act on the following priorities:

- Upgrade all Marimo deployments to version 0.23.0 immediately; all versions through 0.20.4 are affected [11].
- Rotate any credentials or API keys accessible from environments where Marimo was internet-exposed.
- Audit firewall rules to ensure Marimo is not reachable from untrusted networks.
- Treat HuggingFace Spaces URLs as potentially untrusted in the same way other file-hosting services are treated in egress filtering policies.

Background

Marimo is an open-source, reactive Python notebook designed as a next-generation alternative to Jupyter. Unlike Jupyter, Marimo notebooks execute as pure Python scripts, support reactive cell updates, and can be deployed as interactive web applications or shared as self-contained programs [3]. The platform has found adoption across university research labs, startup ML teams, and enterprise data

science environments — registering approximately 20,000 GitHub stars as of April 2026 [11] — particularly among teams building AI-integrated workflows. It integrates natively with AI coding assistants, supports local model execution, and enables notebook-to-app deployment without additional infrastructure.

Marimo can be run in several modes. In its default local mode, the server binds to `localhost` and serves the notebook interface on a local port. However, many deployment scenarios — shared research clusters, containerized environments, Jupyter-replacement servers, and cloud-hosted development boxes — expose Marimo on `0.0.0.0`, making it reachable across the network. The vulnerability disclosed in CVE-2026-39987 specifically affects deployments running in edit mode with any network-accessible binding, which is common in collaborative research and development contexts.

NKAbuse was first documented by Kaspersky researchers in December 2023 as a multiplatform Go-based backdoor that abuses the NKN (New Kind of Network) blockchain protocol for C2 communications [4]. NKN is a legitimate, open-source decentralized networking protocol that uses blockchain architecture to route peer-to-peer network traffic. Because NKN traffic is standard protocol communications rather than connections to attacker-controlled IP addresses, it cannot be blocked using traditional IP or domain blocklisting, requiring behavioral detection instead [4]. The April 2026 campaign represents a new variant of this malware family — previously undocumented — that has been adapted for rapid opportunistic deployment against the AI developer toolchain.

Security Analysis

The Vulnerability: Authentication Bypass on `/terminal/ws`

The technical root cause of CVE-2026-39987 is straightforward: Marimo's WebSocket endpoint at `/terminal/ws`, which provides interactive terminal access to the host system, does not call the `validate_auth()` function before accepting connections. Other WebSocket endpoints in Marimo, including the primary notebook endpoint at `/ws`, correctly invoke authentication validation. The terminal endpoint instead performs only two checks — whether the server is running in a supported mode and whether the platform supports terminal sessions — then opens a full PTY (pseudo-terminal) shell and hands it to the connecting client [2]. Any unauthenticated attacker who can reach the Marimo server over the network can send a single WebSocket connection request to `/terminal/ws` and receive an interactive shell on the underlying host running as the Marimo process user — commonly root in default Docker deployments [11].

The severity of this authentication gap is compounded by what Marimo terminal access exposes. A Marimo server running in an AI development or data science context typically has access to environment files containing database credentials, cloud provider API keys (AWS, GCP, Azure), model API keys (Anthropic, OpenAI, HuggingFace), Git credentials, and internal service tokens. Sysdig's researchers observed that by the third attacker session following initial exploitation, threat actors had located and read `.env` files containing AWS access keys and other application credentials, completing a full credential theft operation in under three minutes [1].

Exploitation Timeline and Scale

The vulnerability was disclosed on April 8, 2026. The first confirmed exploitation attempt was recorded by Sysdig Threat Research at 9 hours and 41 minutes after the public advisory, with no public proof-of-concept exploit code available at that time — suggesting the attacker reconstructed the exploit independently from the advisory description [1][12]. The speed of exploitation suggests threat actors are actively monitoring CVE advisories for AI developer tooling; the specific methods used to develop this exploit are not publicly known [1][12].

The campaign escalated rapidly from initial scanning to multi-stage attacks. Early sessions consisted of simple connection tests verifying that the exploit was functional against reachable targets. Later sessions evolved into multi-hour interactive operations involving reconnaissance, credential extraction, DNS exfiltration, and lateral movement to backend systems including PostgreSQL databases and Redis caches discovered through the leaked environment credentials [1]. The shift from one-shot exploitation to extended interactive access within the same campaign cycle indicates that at least some of the threat actors involved were conducting deliberate, targeted intrusions rather than pure credential-harvesting automation [1][9].

The NKAbuse Variant and HuggingFace Delivery Chain

The most technically significant attack campaign documented during the April 11–14 observation window deployed a new, previously undocumented variant of NKAbuse through an unusual delivery chain. Attackers exploited CVE-2026-39987 to gain shell access, then issued a `curl` command fetching a shell script from a HuggingFace Space named "vsccode-modetx" — a name designed to resemble "vscode-modetx" and evoke an association with the legitimate VS Code development environment [2]. The HuggingFace Space served as a static file host for the dropper script, which in turn downloaded a stripped Go ELF binary called `kagent` — a UPX-packed payload measuring 15.5 MB decompressed [2].

The name "kagent" appears to mirror the legitimate open-source Kubernetes AI agent project of the same name, likely chosen to evade notice in AI infrastructure environments where Kubernetes tooling is in use. By mimicking a tool that is genuinely expected to run as a background service on AI infrastructure, the malware was designed to blend into process and service listings in developer environments. The payload communicates with its C2 infrastructure exclusively over the NKN blockchain network, meaning that no traditional IP-address or domain-based network indicator will identify malicious traffic in isolation.

The dropper script installs persistence through three parallel mechanisms, targeting both Linux and macOS hosts: a systemd user service at `~/.config/systemd/user/kagent.service`, a crontab `@reboot` entry, and a macOS LaunchAgent at `~/Library/LaunchAgents/com.kagent.plist` [2]. All installer output is redirected to `~/.kagent/install.log`, suppressing visible output from standard terminal monitoring. This multi-vector persistence approach reflects an awareness that data science and ML developer machines include both Linux servers and macOS workstations, and the threat actor designed accordingly.

HuggingFace as a Trusted Delivery Vector

The use of HuggingFace Spaces as a malware delivery infrastructure represents a meaningful evolution in attacker technique. HuggingFace is one of the most widely used platforms in the AI development ecosystem, and its domain (`huggingface.co`) is commonly allowlisted in corporate security controls, CDN caching, and developer tool configurations. Security teams that allow outbound connections to AI platforms — as many AI development environments require — will not typically flag a `curl` request to a HuggingFace Space URL as anomalous. This is not a novel attack class; previous campaigns have abused GitHub, Google Drive, and other trusted platforms for payload hosting. However, the specific choice of HuggingFace reflects an attacker population that understands the AI developer threat surface in detail.

This incident also reflects a broader pattern of AI platform infrastructure being exploited for malicious payload hosting. HuggingFace repositories and Spaces have been used in prior campaigns to host malicious ML models and pickle-serialized malware [5]. The platform has implemented security controls over time, but the openness required for its core function of open model and dataset distribution creates an inherent tension with the ability to prevent all malicious use.

Implications for AI Developer Toolchain Security

CVE-2026-39987 is not an isolated incident. It is part of a pattern in which the tools that AI and ML developers use — notebooks, model serving infrastructure, experiment tracking platforms, and data pipeline frameworks — represent a new attack surface that the security industry has not historically monitored with the same rigor applied to enterprise application servers. Marimo is used in environments that are rich with high-value credentials: API keys, model weights, training datasets, cloud provider access, and research intellectual property. The same properties that make these environments productive — network-accessible notebooks, integrated cloud credentials, broad AI platform access — make them lucrative targets.

The exploitation of CVE-2026-39987 within hours of disclosure, by multiple independent threat actors, in the absence of public exploit code, should prompt organizations to revisit their assumption that niche developer tools face lower threat attention than mainstream enterprise software. The evidence from this campaign is the opposite: adversaries are now monitoring CVE advisories for software across categories, including specialized AI development tooling, and are building functional exploits rapidly.

Recommendations

Immediate Actions

Every organization running Marimo should upgrade to version 0.23.0 or later without delay. The vulnerability affects all versions through 0.20.4, and the patch closes the authentication gap by adding the missing `validate_auth()` call to the `/terminal/ws` handler. Upgrade should be treated as an emergency change, not deferred to the next maintenance window, given confirmed active exploitation beginning within hours of disclosure.

Organizations should immediately audit whether any Marimo instances are reachable from untrusted networks. Deployments running with `--host 0.0.0.0` in edit mode are directly exploitable. Access should be restricted to localhost or placed behind an authenticated reverse proxy. Any cloud-hosted Marimo instances should be reviewed for network security group or firewall rules that may permit inbound WebSocket connections from the internet.

All secrets accessible from affected Marimo environments should be rotated immediately, regardless of whether exploitation has been confirmed. Because credential theft can be completed in under three minutes by an attacker with shell access, and because it may leave no obvious artifacts in standard

application logs, absence of a confirmed breach should not be taken as evidence that credentials are unexposed. This rotation should include cloud API keys, database credentials, model API keys, and any tokens stored in `.env` files or shell history on the affected host.

Short-Term Mitigations

For organizations that cannot immediately upgrade, the most effective mitigation is to prevent network access to the Marimo server's WebSocket endpoint. This can be accomplished by binding Marimo to `127.0.0.1` rather than `0.0.0.0`, or by blocking inbound connections to the Marimo port at the network layer. If remote access to Marimo is required, it should be routed through an authenticated proxy or VPN rather than exposed directly. Web application firewalls that can filter WebSocket upgrade requests should be configured to require authenticated sessions before permitting WebSocket connections to internal services.

Organizations should enable logging for all outbound connections from AI development infrastructure, with particular attention to connections initiating file downloads from external platforms including HuggingFace, GitHub, and other model and dataset hosting services. A `curl` or `wget` command downloading an executable from a platform URL and immediately executing it is a high-confidence indicator of a post-exploitation dropper chain in environments where such patterns are not part of approved automation or CI/CD workflows, regardless of the specific platform involved. This logging should be configured to capture the full URL, not merely the domain, to support hunting for the specific "vsccode-modetx" HuggingFace Space and similar typosquatted infrastructure.

Teams should also review whether HuggingFace Spaces and similar AI platform infrastructure should be treated as fully trusted in egress filtering policies, or whether downloaded executables from those platforms should be subject to the same scrutiny as downloads from any other public file-hosting service. Legitimate AI development workflows rarely require fetching and executing binary payloads from HuggingFace Spaces; this pattern is a strong anomaly signal even on platforms that are broadly allowlisted.

Strategic Considerations

The broader implication of CVE-2026-39987 is that AI development toolchain security requires the same disciplined vulnerability management posture that organizations apply to production application infrastructure. Research notebooks, model training environments, and AI development platforms are not lower-risk simply because they are internal or developer-facing tools. They hold highly valuable

credentials and intellectual property, and they are operated by users — data scientists and ML engineers — who are often outside the security team's visibility. The combination of high-value assets and low-visibility operations makes AI developer tooling a priority target for well-resourced threat actors.

Organizations should extend their vulnerability management programs to explicitly include AI development tooling in the software inventory subject to CVE monitoring and patching SLAs. Software like Marimo, Jupyter, LangChain, and similar tools should be treated as production dependencies for the purposes of patch management. Automated scanning and alerting on CVE disclosures for this software category should be configured before the next critical vulnerability, not in response to one.

Finally, the use of blockchain-based C2 in the NKAbuse variant deployed in this campaign warrants attention in detection engineering. Traditional C2 detection relies on identifying outbound connections to known-malicious IP addresses or domains. NKN-based C2 produces legitimate blockchain network traffic that does not match these signatures. Detection of NKAbuse variants requires behavioral indicators — unexpected process execution, novel service installation, suspicious binary names that mimic legitimate tools — rather than network threat intelligence alone. Security teams should review whether their endpoint detection capabilities include these behavioral rules, and specifically whether the persistence mechanisms documented in this campaign (systemd user services in the home directory, crontab reboot entries, macOS LaunchAgents) would trigger alerts in their current configuration.

CSA Resource Alignment

The CSA MAESTRO framework for agentic AI threat modeling provides the most directly applicable analytical lens for this incident. MAESTRO Layer 1 — the AI agent software environment — explicitly identifies the exploitation of vulnerable AI development tools and the compromise of the toolchain runtime as priority threats [6]. CVE-2026-39987 is a canonical Layer 1 attack: it bypasses authentication at the notebook level to gain control of the execution environment, enabling subsequent compromise of credentials, data, and connected systems. MAESTRO's recommended mitigations for this layer — strong authentication on all management interfaces, network segmentation of AI development infrastructure, and monitoring for anomalous process execution — are precisely the controls that would have detected or prevented the NKAbuse deployment chain documented here.

The CSA AI Controls Matrix (AICM) addresses the governance dimensions of AI development toolchain security across several control domains. The Asset Management and Configuration domain provides control objectives for maintaining an authoritative inventory of AI development software and applying timely patches, which is the foundational organizational capability needed to respond to vulnerabilities like CVE-2026-39987 at appropriate speed [7]. The Identity and Access Management domain provides

controls for ensuring that all management interfaces to AI development infrastructure require authentication and enforce least privilege — directly addressing the authentication bypass at the root of this vulnerability. Organizations implementing AICM controls in these domains will have the organizational structure to treat AI developer tooling with the same rigor as production systems.

CSA's Zero Trust guidance is particularly relevant to the network exposure dimension of this vulnerability. A Zero Trust architecture applied to AI development infrastructure would treat Marimo's network-accessible terminal interface as an untrusted surface requiring explicit authentication and authorization regardless of whether the client is on a corporate network. Under Zero Trust principles, the missing `validate_auth()` call in Marimo's `/terminal/ws` handler would represent a policy violation against the network's access control requirements, not merely a software defect [8]. Organizations adopting Zero Trust for AI infrastructure should verify that all notebook and development platform management interfaces are behind authenticated access controls as a Zero Trust policy requirement, independent of the individual software's own authentication implementation.

The CSA STAR program provides the assurance mechanism for evaluating AI development platform security claims. When procuring or adopting new AI development tools — notebooks, model serving platforms, experiment trackers, or pipeline frameworks — security teams should request STAR-level attestation or equivalent vendor security documentation that specifically addresses the security architecture of any network-accessible management interfaces. For open-source tools such as Marimo, reviewing security architecture documentation, recent CVE history, and the project's responsible disclosure practices serves as a comparable proxy for the assurance STAR attestation provides for commercial vendors. The rapid exploitation timeline in this incident illustrates the cost of deploying AI development tooling without vetting the security of its network-facing attack surface.

References

[1] Sysdig Threat Research Team. "Marimo OSS Python Notebook RCE: From Disclosure to Exploitation in Under 10 Hours." Sysdig, April 2026.

[2] Sysdig Threat Research Team. "CVE-2026-39987 Update: How Attackers Weaponized Marimo to Deploy a Blockchain Botnet via HuggingFace." Sysdig, April 2026.

[3] Marimo. "marimo: A Reactive Notebook for Python." marimo.io, 2024.

[4] Kaspersky Global Research & Analysis Team. "Unveiling NKAbuse: A New Multiplatform Threat Abusing the NKN Protocol." Securelist, December 2023.

[5] ReversingLabs. "Malicious ML Models Discovered on Hugging Face Platform." ReversingLabs Blog, February 2025.

[6] Cloud Security Alliance. "Agentic AI Threat Modeling Framework: MAESTRO." CSA, February 2025.

[7] Cloud Security Alliance. "AI Controls Matrix." CSA, 2025.

[8] Cloud Security Alliance. "Zero Trust Advancement Center." CSA, 2024.

[9] BleepingComputer. "Hackers Exploit Marimo Flaw to Deploy NKAbuse Malware from Hugging Face." BleepingComputer, April 2026.

[10] CyberSecurityNews. "Attackers Weaponize CVE-2026-39987 to Spread Blockchain-Based Backdoor Via Hugging Face." CyberSecurityNews, April 2026.

[11] Endor Labs. "Root in One Request: Marimo's Critical Pre-Auth RCE (CVE-2026-39987)." Endor Labs Blog, April 2026.

[12] The Hacker News. "Marimo RCE Flaw CVE-2026-39987 Exploited Within 10 Hours of Disclosure." The Hacker News, April 2026.

