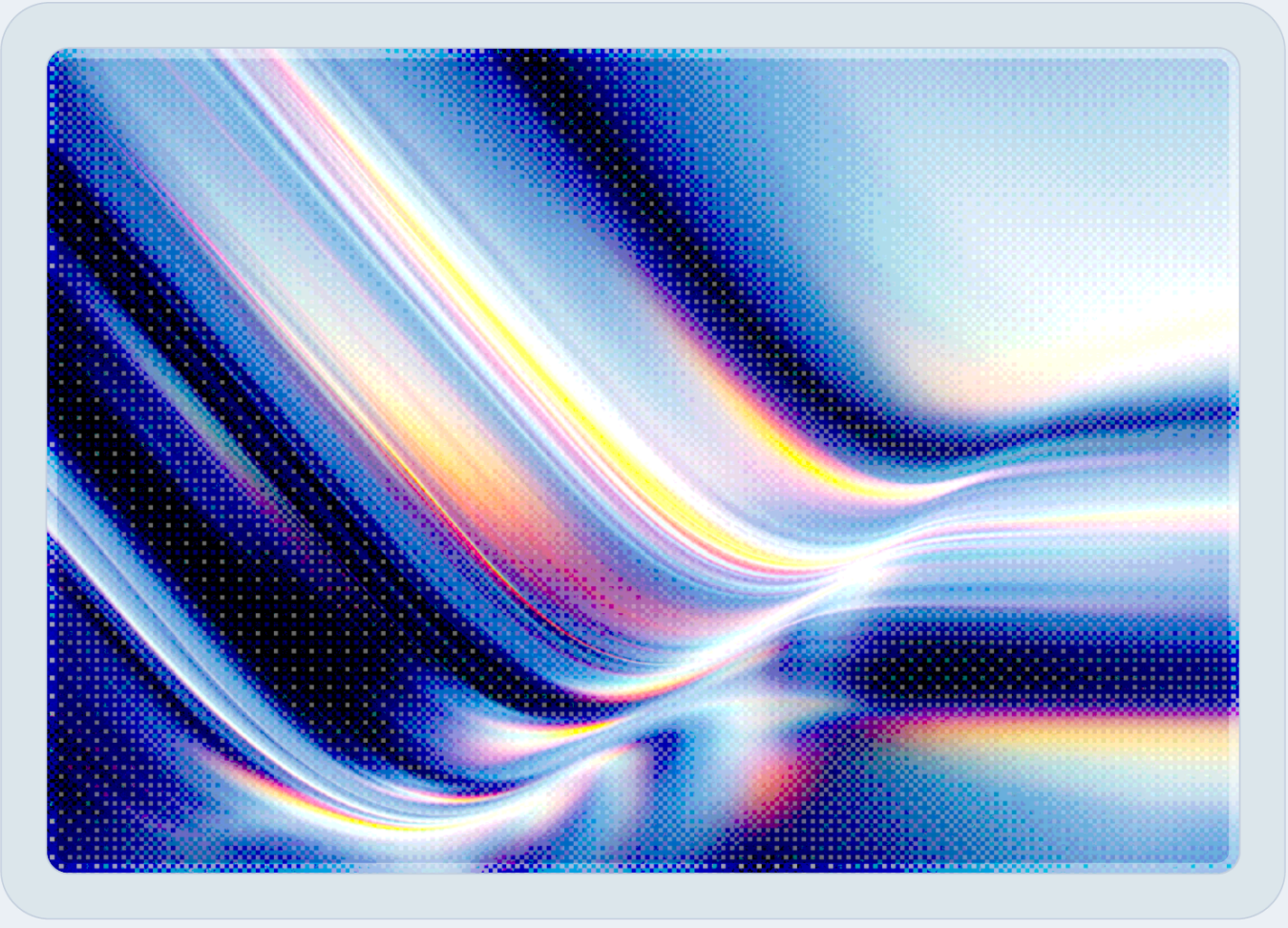
CSAI Foundation | Cloud Security Alliance

PraisonAI Auth Bypass: Zero Hours to Exploitation

CVE-2026-44338 and the Accelerating Attack Surface of Agentic AI Frameworks

2026-05-16

AI-assisted Rapid Research



CSAI

cloud security alliance®

Key Takeaways

- CVE-2026-44338 (CVSS 7.3), a missing-authentication vulnerability in PraisonAI versions 2.5.6 through 4.6.33, was disclosed on May 11, 2026; automated scanners were probing internet-exposed `/agents` and `/chat` endpoints within 3 hours and 44 minutes of the public advisory [1][2][15].
- The root cause is architectural: a legacy Flask-based API server in PraisonAI's codebase hard-codes `AUTH_ENABLED = False` and `AUTH_TOKEN = None`, allowing any network-accessible caller to enumerate agent configurations and trigger unauthenticated workflow execution [3][4][7].
- Because PraisonAI agents execute real tools — LLM provider calls, file system access, API integrations, and code execution — an auth bypass in this context is not a data exposure event; it is an arbitrary capability invocation event, with blast radius (the scope of potential impact) bounded only by the tools each agent has been granted [5].
- CVE-2026-44338 is part of a documented surge: more than 30 CVEs were filed against agentic AI tooling in a 60-day window in early 2026, strongly suggesting that AI framework attack surfaces are being instrumented at machine speed by parties capable of deploying purpose-built scanning infrastructure within hours of disclosure [2][6][16].
- Affected organizations should update to PraisonAI version 4.6.34 or later immediately, audit API server logs for unauthorized workflow invocations, and rotate all credentials referenced in `agents.yaml` configurations.

Background

What PraisonAI Is

PraisonAI is an open-source multi-agent orchestration framework designed to simplify the development and production deployment of autonomous AI systems. Developed by MervinPraison and published on GitHub with approximately 7,100 stars, the project integrates foundational orchestration logic from AutoGen and CrewAI into a low-code interface that enables developers to configure multi-agent workflows in as few as five lines of code [5]. Its feature set includes built-in memory, retrieval-augmented

generation (RAG) support, and compatibility with more than 100 large language model providers, making it a frequently referenced starting point for developers building agentic AI capabilities. The framework is available in both Python and TypeScript, is marketed for production deployment, and supports connecting AI agents to real tools, external APIs, cloud services, and data stores.

PraisonAI occupies a representative position in the current AI tooling ecosystem: it is not an experimental research library, but a production-grade orchestration framework that routes agent actions into real-world infrastructure. This distinction matters for security analysis. When an orchestration framework ships with authentication disabled by default, the affected population is not a narrow set of researchers running local experiments — it is the broader set of organizations that adopted it for production use and reasonably expected that a production-grade framework had been designed with secure defaults.

CVE-2026-44338: The Authentication Bypass

CVE-2026-44338 was published to the National Vulnerability Database on May 11, 2026 with a CVSS base score of 7.3 and classified under CWE-306: Missing Authentication for Critical Function [1]. The affected versions span PraisonAI 2.5.6 through 4.6.33. The vulnerability resides in a legacy Flask-based API server (`src/praisonai/api_server.py`) that was included in production release builds across this version range. Within that server, two configuration parameters combine to nullify the authentication mechanism: `AUTH_ENABLED` is set to `False` as its default value, and `AUTH_TOKEN` is set to `None` [3][4][7]. When this server is running and network-reachable, no credential check occurs on any incoming request.

The specific endpoints exposed by this configuration gap are the `/agents` endpoint, which returns the configured agent definitions from the instance's `agents.yaml` file, and the `/chat` endpoint, which accepts a message and triggers the execution of the configured agent workflow. An unauthenticated caller who can reach either endpoint can enumerate the agents deployed on the instance and invoke them at will. The fixed version, 4.6.34, replaces the legacy server with a `serve agents` command that binds locally to `127.0.0.1` by default and requires an explicit API key to be passed at startup — a design that fails safely when the API key is absent rather than permitting open access [4][7].

Disclosure and Exploitation Timeline

The exploitation timeline for CVE-2026-44338 is the most operationally significant aspect of this case. Sysdig's threat research team documented that a scanner labeled "CVE-Detector/1.0" was probing internet-accessible PraisonAI API server ports for this specific vulnerability within 3 hours and 44

minutes of the CVE's public disclosure [2][15]. This behavior — purpose-built scanners targeting newly disclosed AI framework vulnerabilities within hours of publication — is consistent with a pattern documented across multiple recent AI framework CVEs, not an isolated incident [15][16]. The deployment speed and specificity of the scanner tooling indicates that parties targeting AI framework infrastructure are organized, automated, and purpose-built to instrument the AI tooling attack surface as disclosures emerge. The speed of this response eliminates any practical patch-before-exploit window for organizations that do not maintain near-real-time vulnerability monitoring and patch deployment capabilities for their AI tooling inventory.

Security Analysis

Authentication Defaults as a Systemic Risk in AI Tooling

The authentication failure in PraisonAI's legacy API server reflects a structural tension that pervades open-source AI framework development. Projects optimized for developer experience — and PraisonAI explicitly prioritizes the ability to deploy a working multi-agent system in five lines of code — frequently defer authentication configuration to the user rather than enforcing it by default. This design philosophy is understandable in the context of rapid prototyping: authentication setup introduces friction, and most early adopters of a new framework are experimenting on local machines with no network exposure. The problem arises when the same configuration that was optimized for frictionless experimentation ships unchanged into production releases used by enterprises that are not experimenting.

What distinguishes this case from a typical insecure-default vulnerability is the maturity of the deployment context. PraisonAI is not a beta library; it is a framework positioned for production multi-agent deployments, integrated with LLM providers and real tool infrastructure. The gap between the framework's production-readiness marketing and its authentication posture is the security debt that CVE-2026-44338 makes concrete. Organizations that adopt AI frameworks based on feature capabilities — agent memory, RAG integration, multi-LLM support — without evaluating authentication defaults are accepting an exposure they may not understand until a scanner finds their instance.

The Agentic Blast Radius of an Authentication Bypass

The security implications of an unauthenticated API in an agentic context differ fundamentally from those of a traditional web application API. When a conventional REST endpoint lacks authentication, the typical impact is unauthorized data access or modification of the application's own resources. When an

agentic orchestration framework's endpoint lacks authentication, the impact is unauthorized capability execution — and the capability set is bounded not by what the API knows, but by what the agents are permitted to do.

PraisonAI agents operate by invoking tools: functions that interact with external systems, execute code, read and write files, make HTTP requests, call LLM providers with billing implications, and manipulate cloud APIs [5]. An attacker who reaches the unauthenticated `/chat` endpoint on an internet-exposed PraisonAI instance can send arbitrary messages to configured agents and receive the results of their tool executions. Depending on the specific configuration of the instance, this may enable exfiltration of data processed by the agents, consumption of LLM provider credits under the victim organization's billing account, lateral movement into systems the agents are authorized to access, and manipulation of downstream workflows connected to agent outputs. The `agents.yaml` configuration file, which defines agent tool bindings and may include API keys and service credentials inline, represents a direct credential exposure surface and should be treated as a rotation priority for any instance that was network-reachable during the vulnerability window [4][7].

CSA survey research published in 2026 found that 74% of security professionals agree that AI agents in their organizations routinely receive excessive access relative to their operational requirements, and that 81% agree prompt manipulation could cause agents to disclose credentials to unauthorized parties [8]. CVE-2026-44338 demonstrates the direct exploitation path for both: an unauthenticated network caller who can invoke the `/chat` endpoint can prompt agents directly, without the prompt manipulation layer, and potentially access any data or capabilities the agents are authorized to reach.

Zero-Window Exploitation and the AI Framework Threat Landscape

The three-hour-and-forty-four-minute time-to-scan for CVE-2026-44338 places AI framework vulnerabilities firmly within the zero-window exploitation environment — where exploitation begins before any practical patch-before-exploit window can open — that previously characterized only the most critical enterprise software vulnerabilities: web application servers, VPN appliances, and firewall management interfaces. This convergence reflects two parallel developments. First, AI frameworks have achieved sufficient production deployment density that they are economically attractive targets for automated scanning campaigns: a successful exploit against an internet-exposed agentic orchestration instance yields access to LLM provider credentials, connected service credentials, and potentially sensitive data being processed by agents, all in a single unauthenticated request. Second, threat actors are actively investing in AI-specific scanning and exploitation tooling, as evidenced by the purpose-built "CVE-Detector/1.0" scanner observed by Sysdig targeting PraisonAI specifically [2].

CVE-2026-44338 does not stand alone. The 60-day window preceding its disclosure saw more than 30 CVE filings against agentic AI tooling across multiple frameworks, spanning a range of severity levels [2][6][16]. Notable high-severity examples include Langflow CVE-2026-33017 (CVSS 9.8), a remote code execution vulnerability in a popular low-code AI application builder exploited in the wild within 20 hours of disclosure [2][17], and a cluster of vulnerabilities affecting Microsoft Semantic Kernel, another widely deployed orchestration framework [6]. Microsoft's security research team has documented the mechanism by which agentic frameworks transform prompt injection from a content manipulation technique into a remote code execution primitive once agents are wired to real tools [6]. CISA and its Five Eyes counterparts published formal joint guidance in May 2026 on the careful adoption of agentic AI services, identifying expanded attack surface, privilege creep, behavioral misalignment, and obscure audit trails as the four primary risk categories requiring active management [9]. The convergence of rapid CVE disclosure cadence, near-instantaneous automated scanning, and formal government advisory guidance signals that the AI framework threat environment has matured past the early-adoption period in which security consequences were largely theoretical.

One dimension of this threat landscape that receives less attention is the shadow deployment problem. PraisonAI's five-line deployment model makes it straightforward for instances to spin up in development environments, CI/CD pipelines, and engineer workstations with minimal formal review, and without being added to asset inventories or formally decommissioned when no longer actively used. CSA's 2026 survey on AI agent governance found that 82% of organizations discovered previously unknown AI agents in their environments despite believing they had high visibility [10]. An organization may have no patched PraisonAI instances in its production environment and multiple unpatched instances in development environments it did not know were network-accessible.

Recommendations

Immediate Actions

Organizations should update all PraisonAI deployments to version 4.6.34 or later without delay. The update is the authoritative remediation; the new `serve agents` command corrects the insecure-by-default behavior that is the root cause of CVE-2026-44338. Prior to or alongside patching, operators should audit API server logs for requests to the `/agents` and `/chat` endpoints originating from unexpected source addresses, and should treat any unexpected access as a potential indicator of unauthorized workflow execution. Discovery and inventory are a prerequisite for this audit: organizations

should identify all PraisonAI instances across development, staging, and production environments, including any that may have been deployed informally by individual engineers without a formal provisioning process.

Any credentials referenced in `agents.yaml` files on potentially exposed instances should be rotated immediately. This includes LLM provider API keys, cloud service credentials, OAuth tokens, and any other secrets the agents use to interact with external systems. The rotation should be completed before investigators determine whether exploitation occurred; waiting for confirmation of exploitation before rotating credentials allows the exposure window to remain open.

Short-Term Mitigations

For organizations that cannot immediately update, network-level controls provide meaningful risk reduction. PraisonAI API server instances should not be reachable from the public internet unless that exposure is operationally required and intentional. Firewall rules, security group policies, or reverse-proxy authentication layers placed in front of the legacy API server substantially limit the population of potential exploiters. Instances that serve only local developer workflows should be confirmed to bind to loopback addresses and should not be reachable from other hosts on the network.

Organizations should conduct a broader inventory of agentic AI tools deployed across their environments and apply the same exposure assessment applied here to other frameworks that ship API servers. The insecure-default pattern seen in CVE-2026-44338 is not unique to PraisonAI; frameworks built on AutoGen, CrewAI, LangChain, and similar orchestration libraries may carry analogous default configurations that have not yet received CVE-level attention. Reviewing authentication defaults, API binding addresses, and credential storage patterns for each deployed AI framework is an appropriate response to a threat environment in which more than 30 AI framework CVEs were filed in 60 days.

Strategic Considerations

The PraisonAI case makes a compelling argument for treating AI framework components as a distinct category within vulnerability management programs, with monitoring cadence and patching SLAs comparable to those applied to web application servers and API gateways. The exploitation timeline observed for CVE-2026-44338 — active scanning within hours of disclosure — is incompatible with patch cycles measured in weeks. Organizations whose vulnerability management programs cover enterprise software platforms but do not explicitly inventory and monitor open-source AI framework dependencies should close that gap as a near-term priority.

Security teams evaluating new AI frameworks for adoption should assess secure-by-default configuration as a first-class evaluation criterion alongside feature capabilities. A framework that ships authentication-off requires operators to make an active configuration decision to secure it; a framework that ships authentication-on and requires an active decision to open it is demonstrably safer under realistic operational conditions, since it ensures authentication is enforced in the absence of explicit operator action rather than requiring operators to take a hardening step that may be omitted. Procurement checklists and security review templates for AI tooling adoption should include explicit questions about API server default binding addresses, default authentication state, and the behavior of the server when no API key has been configured.

CSA Resource Alignment

MAESTRO

CSA's MAESTRO threat modeling framework for agentic AI provides directly applicable analytical structure for understanding the risks exposed by CVE-2026-44338 [11]. The missing authentication in PraisonAI's API server maps to MAESTRO's Layer 3 (Agent Frameworks) — the layer that encompasses the agent reasoning loop, tool dispatch logic, and the execution environment in which agent actions are carried out. MAESTRO explicitly identifies authentication bypass and unauthenticated workflow invocation as Layer 3 threat patterns, and it documents how Layer 3 compromise enables downstream consequences at Layer 2 (Data and Context) and Layer 1 (Foundation Models) when agents are triggered to process, exfiltrate, or manipulate data accessible through their tool bindings.

The broader zero-window exploitation pattern — automated scanning and targeting of newly disclosed AI framework vulnerabilities — maps to MAESTRO's supply chain and framework integrity threat categories. Security teams using MAESTRO to model their agentic AI deployments should explicitly assess whether any deployed AI framework ships with API servers whose authentication state is configurable, and whether the default state of those servers is secure or open. MAESTRO's compound threat analysis is particularly relevant to scenarios where an attacker who gains unauthenticated API access subsequently uses that access to inject prompts that cause agents to disclose credentials or execute actions enabling lateral movement.

AI Controls Matrix (AICM)

CSA's AI Controls Matrix (AICM) addresses the governance and technical control requirements for AI systems across the AI supply chain, covering authentication, access management, vulnerability management, and incident response within AI-specific deployment contexts [12]. CVE-2026-44338 is a concrete instance of control failure in the authentication and access management domains: the framework lacked effective authentication for a critical execution function, and the absence of secure default configuration allowed that failure to propagate into production deployments where operators had not taken explicit hardening steps.

Organizations mapping their agentic AI framework deployments against the AICM should ensure that their authentication and access controls for AI framework APIs are assessed as active security controls subject to testing and monitoring — not assumed capabilities inherited from the framework vendor. The AICM's guidance on shared security responsibility is directly applicable here: when an organization deploys an open-source framework, it inherits the framework's security properties and cannot rely on the upstream project to enforce security configurations that the framework itself leaves to the operator. The AICM control language for authentication, access governance, and vulnerability response should be applied to AI framework API surfaces with the same rigor applied to cloud service APIs and application endpoints.

Agentic AI Security Research

CSA's Agentic AI Red Teaming Guide documents the discovery and exploitation techniques used against AI agent infrastructure, including API enumeration, workflow injection, and credential harvesting through agent-mediated tool calls [13]. Its structured test cases are directly applicable to post-incident assessment of potentially exposed PraisonAI deployments, and its coverage of authorization testing provides a methodology for evaluating whether other AI frameworks in an organization's environment carry analogous authentication weaknesses.

The broader governance picture is documented in CSA's 2026 AI agent survey series, which found that only 31% of organizations have formally adopted AI agent governance policies and that 47% have experienced AI agent security incidents in the past twelve months [14]. CVE-2026-44338 illustrates the causal pathway between governance gaps and security incidents: when AI frameworks are adopted without formal security review, deployed without hardening steps, and operated without continuous monitoring, the result is exactly the exposure profile that automated scanners are now purpose-built to discover and exploit. The survey data and the CVE are two data points on the same curve.

References

[1] NIST National Vulnerability Database. "CVE-2026-44338 Detail." NVD, May 11, 2026.

[2] Sysdig Threat Research Team. "CVE-2026-44338: PraisonAI Authentication Bypass in Under 4 Hours and the Growing Trend of Rapid Exploitation." Sysdig, May 2026.

[3] CVEFeed. "CVE-2026-44338 – PraisonAI ships and generates a legacy API server with authentication disabled." CVEFeed, May 2026.

[4] Snyk Security. "Missing Authentication for Critical Function in praisonai (SNYK-PYTHON-PRAISONAI-16635978)." Snyk, May 2026.

[5] MervinPraison. "PraisonAI – Open Source Multi-Agent Orchestration Framework." GitHub, 2026.

[6] Microsoft Security Blog. "When Prompts Become Shells: RCE Vulnerabilities in AI Agent Frameworks." Microsoft, May 7, 2026.

[7] The Hacker News. "PraisonAI CVE-2026-44338: Auth Bypass Targeted Within Hours of Disclosure." The Hacker News, May 2026.

[8] Cloud Security Alliance. "Identity and Access Gaps in the Age of Autonomous AI." CSA, 2026.

[9] CISA and International Partners. "Careful Adoption of Agentic AI Services." CISA, May 2026.

[10] Cloud Security Alliance. "Autonomous but Not Controlled: AI Agent Incidents Now Common in Enterprises." CSA, 2026.

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

[12] Cloud Security Alliance. "AI Controls Matrix (AICM)." CSA, 2025.

[13] Cloud Security Alliance. "Agentic AI Red Teaming Guide." CSA, 2026.

[14] Cloud Security Alliance. "Enterprise AI Security Starts with AI Agents: State of AI Agents Security Survey Report." CSA, 2026.

[15] CSO Online. "PraisonAI Vulnerability Gets Scanned Within 4 Hours of Disclosure." CSO Online, May 2026.

[16] Cloud Security Alliance. "Agentic AI Framework CVEs Under Active Exploitation." CSA Labs, March 28, 2026.

[17] NIST National Vulnerability Database. "CVE-2026-33017 Detail." NVD, 2026.