CVE-2026-33017: Unauthenticated RCE in Langflow AI Pipelines

Unofficial AI-assisted Research

Cloud Security Alliance AI Safety Initiative

2026-03-30

Key Takeaways

CVE-2026-33017 is a critical unauthenticated remote code execution (RCE) vulnerability in Langflow, the open-source AI workflow orchestration platform developed by DataStax. With CVSS scores of 9.8 (NVD/v3.1) and 9.3 (GitHub CNA/v4.0) [1][2], the flaw allows any unauthenticated network attacker to execute arbitrary operating system commands on a Langflow server by supplying attacker-controlled workflow data to an endpoint designed for public flow access. Exploitation was observed in the wild within 20 hours of disclosure [3][13], CISA added the vulnerability to its Known Exploited Vulnerabilities (KEV) catalog on March 25, 2026, with a federal agency remediation deadline of April 8, 2026 [4]. Organizations running Langflow versions 1.8.2 or earlier must upgrade to version 1.9.0 immediately [1][2]. This is the second critical unauthenticated `exec()`-based RCE in Langflow to reach the CISA KEV catalog, following CVE-2025-3248 in May 2025 [5], indicating a structural security pattern in the platform that warrants architectural scrutiny beyond simple patching.

Background

Langflow and the AI Orchestration Landscape

Langflow is an open-source, low-code visual builder for AI agent workflows and retrieval-augmented generation (RAG) pipelines. Originally a community project, it was acquired by DataStax in April 2024 and is now offered both as a self-hosted solution and as Langflow Cloud, a managed service integrated with DataStax's platform — itself being integrated into IBM's watsonx AI portfolio [6]. The platform has accumulated over 100,000 GitHub stars, reflecting broad adoption among developers building AI-powered applications [15]. Note that DataStax announced the April 9, 2026 shutdown of Langflow Cloud [6]; the vulnerability discussed in this note applies to self-hosted instances, which remain in active deployment.

The platform's value proposition is its drag-and-drop canvas interface, where each node represents a component — a large language model, a tool integration, a data source, or an output destination. Built on top of LangChain, it enables developers to wire together LLMs, vector databases, external APIs, and custom Python code without writing a complete application. This architecture is a primary factor in Langflow's value as an attack target: production deployments often store AI provider credentials — API

keys for services such as OpenAI, Anthropic, and AWS — along with database connection strings and downstream service secrets, as documented by the credential harvesting activity observed during exploitation of this vulnerability [3]. A single compromised instance can therefore provide lateral access across an organization's entire AI infrastructure.

Langflow's default deployment configuration compounds this risk. When installed via the standard `pip install langflow` path, the platform starts with `AUTO_LOGIN=true`, meaning an unauthenticated user can obtain a superuser token immediately without credentials [2]. This default, combined with the vulnerability described below, effectively eliminated even the minimal access barrier of knowing a public flow's UUID.

The Public Flow Architecture and Its Security Trade-off

Langflow supports "public flows" — shared workflows that end users can invoke without logging in, enabling use cases like embedded chatbots and public-facing AI applications. To support this pattern, the platform exposes a purpose-built unauthenticated HTTP endpoint: `POST /api/v1/build_public_tmp/{flow_id}/flow`. This endpoint was designed to execute a stored workflow identified by its UUID, accepting a `client_id` cookie to track session state. The endpoint architecture reflected a reasonable design goal — enabling unauthenticated public access to specific shared flows — but the implementation failed to isolate an attacker-controllable parameter from the downstream execution path. Specifically, the endpoint accepted an optional `data` parameter that, when present, substituted the attacker's payload in place of the stored workflow definition — and that substituted data flowed directly into an unsandboxed Python `exec()` call.

Security Analysis

Technical Root Cause

The vulnerability is classified under CWE-94 (Improper Control of Generation of Code), CWE-95 (Eval Injection), and CWE-306 (Missing Authentication for Critical Function) [1]. The affected code resides in `src/backend/base/langflow/api/v1/chat.py`, with the ultimate execution landing in `src/lfx/src/lfx/custom/validate.py` around line 397, where `exec(compiled_code, exec_globals)` processes component definitions with no sandboxing, no AST-level filtering of dangerous built-ins, and no authentication guard [2].

The exploitation path traverses ten call-chain layers from the HTTP endpoint to the `exec()` call. The critical insight is that attacker-supplied Python code embedded in node definitions executes during graph *construction* — before flow execution even begins. An assignment statement such as `_x = __import__('os').popen('id').read()` triggers immediately when Langflow builds the attacker-supplied workflow object. This design means that defenses applied at the execution stage — rather than at graph construction — are ineffective against this specific attack path, as the injected code runs before execution controls are evaluated.

When `AUTO_LOGIN=true` (the default), no public flow UUID is even required: an unauthenticated attacker can first obtain a superuser token through the auto-login mechanism, create a public flow, and then exploit it — removing the only meaningful access barrier in the original design [2]. The fix in Langflow 1.9.0 removes the injectable `data` parameter from the endpoint entirely, eliminating this unauthenticated code injection pathway [2].

Exploitation Timeline and Threat Actor Activity

Sysdig's threat research team observed the first exploitation attempt against honeypot infrastructure at 16:04 UTC on March 18, 2026 — approximately 20 hours after the advisory was published [3]. Multiple attacker types were active within that initial 24-hour window.

Sysdig's analysis identified two categories of attacker activity within the initial window [3]. The first involved automated scanning using publicly available nuclei templates, probing with basic payloads such as `__import__('os').popen('id').read()` and exfiltrating results via interactsh out-of-band callback infrastructure. The second, operating from DigitalOcean infrastructure, used custom Python exploit tooling with pre-staged command-and-control servers and conducted methodical credential harvesting — enumerating environment variables, searching for `.env` files and SQLite databases, and targeting the specific secrets known to reside in Langflow deployments: API keys for OpenAI, Anthropic, and AWS, database connection strings, and application secrets [3]. The presence of pre-staged infrastructure is consistent with either rapid post-disclosure preparation or prior knowledge of the vulnerability before public disclosure — a distinction Sysdig's analysis does not resolve definitively.

Public proof-of-concept exploit repositories appeared on GitHub within hours of disclosure [7][8]. CISA's KEV addition on March 25, 2026 confirms active exploitation beyond honeypot observations [4], and the agency's mandate for federal remediation by April 8 indicates an assessment of operationally significant risk.

The Recurring `exec()` Pattern in Langflow

CVE-2026-33017 is not an isolated incident; it is the latest expression of a structural vulnerability pattern in Langflow's architecture — the CSA AI Safety Initiative's assessment based on the observable history of repeated exploitation through the same underlying mechanism. The platform's design relies on dynamic code execution — `exec()` — to handle custom Python components, and this capability has been repeatedly accessible to unauthenticated or minimally authorized users through different endpoints across multiple releases.

CVE-2025-3248 (CVSS 9.8), added to the CISA KEV catalog in May 2025, exploited the same underlying `exec()` path through the `/api/v1/validate/code` endpoint [5][12]. The remediation for that vulnerability added authentication to the code validation endpoint — but the identical dangerous code path remained reachable through the unauthenticated public flow build endpoint, which could not simply require login without breaking its intended public functionality. The result was a partial fix that closed one door while leaving another open, reflecting the inherent difficulty of securing a dynamic code execution engine that must simultaneously support both authenticated and unauthenticated use cases. CVE-2025-34291 (CVSS 9.4), also targeting Langflow in 2025, chained a CORS misconfiguration with the code execution path for an alternative account takeover and RCE vector [9].

This pattern suggests that the underlying architectural risk — a Python `exec()` call reachable via HTTP without sandboxing — may persist in future releases unless DataStax undertakes a more fundamental redesign of the custom component execution model, potentially adopting AST-level code analysis, a restricted execution namespace, or a subprocess-based isolation boundary for user-supplied code.

Recommendations

Immediate Actions

Organizations running any version of Langflow up to and including 1.8.2 (or development releases up to 1.9.0.dev11) should treat this as an emergency patching event and upgrade to Langflow 1.9.0 immediately [1][2]. Note that some secondary sources cite 1.8.1 as the affected version boundary; NVD and the GitHub Security Advisory both specify 1.8.2, and upgrading to 1.9.0 is the definitive remediation regardless of this ambiguity. The CISA KEV deadline for federal agencies is April 8, 2026, but given the active exploitation timeline, commercial organizations should not wait for that date. Any Langflow instance that was network-accessible and unpatched between March 17 and the date of patching should

be treated as potentially compromised and undergo incident response procedures: rotate all API keys and credentials stored in the environment, audit outbound network connections for data exfiltration to unknown destinations, and review environment variable contents and `.env` files for evidence of unauthorized access.

For organizations unable to patch immediately, the highest-priority interim mitigation is to place Langflow behind an authenticated reverse proxy or VPN gateway, ensuring that the `/api/v1/build_public_tmp/` endpoint is not reachable from untrusted networks. Administrators should also set `AUTO_LOGIN=false` in production configurations and disable the public flows functionality if not required for operational use cases.

Network-level indicators of compromise include outbound connections to interactsh callback domains (`.oast.live`, `.oast.me`, `.oast.pro`, `.oast.fun`) and known attacker infrastructure including the DigitalOcean ranges associated with the March 2026 campaign [3]. At the detection layer, Qualys customers can use QID 733892 to identify vulnerable instances [10].

Short-Term Mitigations

Beyond emergency patching, organizations should audit their Langflow deployment configurations to remove the insecure defaults that amplify this vulnerability class. Specifically, `AUTO_LOGIN=true` should never be present in production deployments; this setting was designed for local development convenience and its presence in internet-facing configurations is a critical misconfiguration independent of any specific CVE. Network segmentation policies should prevent Langflow servers from making outbound connections to the internet except to explicitly permitted AI provider endpoints, limiting the blast radius of any future credential harvesting.

For organizations that deploy Langflow as part of an AI pipeline serving production workloads, the credential storage pattern deserves attention as a risk in its own right. Langflow instances that hold API keys for multiple AI providers and cloud platforms create a high-value aggregation point; compromising a single Langflow server can yield credentials enabling lateral access to downstream AI services, cloud infrastructure, and data stores that may represent significantly higher business risk than the Langflow instance itself. Consider externalizing credentials to a secrets management system (e.g., HashiCorp Vault, AWS Secrets Manager) with per-request retrieval rather than persisting them in Langflow's environment.

Strategic Considerations

The recurring RCE pattern in Langflow reflects a broader design tension in AI orchestration platforms: the need to support arbitrary user-defined components and integrations inherently requires executing user-supplied code, and building a truly secure execution sandbox for Python is a non-trivial engineering challenge. Security and engineering teams evaluating or deploying AI orchestration platforms should assess whether the platform uses `exec()` or `eval()` for dynamic component loading, how that execution is isolated from the host operating system, and what authentication controls govern access to code execution endpoints.

For enterprises with significant Langflow deployments, the structural pattern of repeated critical RCEs may warrant a vendor engagement discussion with DataStax about their secure development lifecycle for the platform — specifically, whether a third-party security audit of the custom component execution architecture has been conducted. The Barrack AI analysis of CVE-2026-33017 demonstrates that the March 2026 fix addressed the immediate injection vector but left the underlying `exec()` architecture unchanged; a future re-opening of this attack surface through a different pathway remains plausible [11].

At the portfolio level, organizations should inventory all AI orchestration and workflow tools deployed across teams, as Langflow is representative of a category of AI orchestration platforms that may combine internet-facing APIs with dynamic component execution and stored AI provider credentials — a profile that applies to similar workflow automation and AI pipeline tools across the market. This combination of capabilities creates a high-risk architecture profile that requires consistent network isolation, secrets management, and authentication controls to deploy safely, regardless of which specific product is in use.

CSA Resource Alignment

MAESTRO: Threat Modeling for Agentic AI Systems

CVE-2026-33017 is directly relevant to the CSA MAESTRO framework for agentic AI threat modeling. Langflow functions as an AI orchestration layer — precisely the architecture MAESTRO analyzes for trust boundary violations and lateral movement risks. The exploitation path in this vulnerability follows what MAESTRO characterizes as Layer 5 (Tool and Integration Ecosystem) and Layer 2 (Data Operations) threats: an attacker exploits an external-facing orchestration endpoint to inject arbitrary code that executes with the platform's full runtime privileges, potentially enabling access to all connected tools,

data sources, and downstream AI providers. Organizations should use MAESTRO's threat modeling methodology to assess the trust architecture of their Langflow deployments, particularly the boundaries between public-facing flows and authenticated administrative functions.

AI Controls Matrix (AICM) Alignment

The AICM, CSA's primary framework for AI security controls and a superset of the CCM, maps directly to the control failures exposed by this vulnerability. The absence of authentication on the public flow build endpoint reflects failures in identity and access management controls (AICM identity domains), while the unsandboxed `exec()` execution reflects failures in workload integrity and application security controls. Organizations implementing AICM should map CVE-2026-33017 to their control assessments for AI pipeline components, particularly controls governing code execution isolation, API authentication requirements, and secrets management for AI service credentials.

AI Organizational Responsibilities

The CSA AI Organizational Responsibilities series — covering governance, core security responsibilities, and AI tools and applications — provides the accountability framework for the organizational failures that lead to vulnerabilities like this being exploited at scale. The widespread default use of `AUTO_LOGIN=true` in production deployments and the failure to place Langflow behind authenticated proxies reflect gaps in AI tool procurement and configuration governance processes. Organizations should incorporate AI orchestration platforms into their AI tool risk assessment processes as defined in the AI Organizational Responsibilities framework, treating them as high-risk components due to their combination of credential aggregation, code execution, and internet-facing APIs.

Agentic AI Red Teaming Guide

The CSA Agentic AI Red Teaming Guide (2025) identifies supply chain and dependency attacks, agent authorization hijacking, and multi-agent exploitation as among the twelve critical vulnerability categories for agentic AI systems. CVE-2026-33017 is a real-world instantiation of the authorization hijacking category: an attacker bypasses all agent authorization controls by exploiting the underlying orchestration platform rather than the agent logic itself. Security teams should include AI orchestration platforms in their red team scope using the methodologies described in the Guide, treating platforms like Langflow as privileged infrastructure components rather than simple developer tools.

STAR for AI Registry

Organizations deploying Langflow as part of an AI service offering to customers or partners should consider the implications of CVE-2026-33017 for their STAR for AI Level 1 self-assessment (AI-CAIQ) submissions, particularly disclosures related to infrastructure security, patch management, and secrets handling. The STAR Registry provides a mechanism for transparent communication of AI security posture; timely remediation and accurate AI-CAIQ responses regarding this vulnerability class support the trust objectives the program is designed to advance.

References

[1] NIST National Vulnerability Database, "CVE-2026-33017 Detail," NVD, March 20, 2026. https://nvd.nist.gov/vuln/detail/CVE-2026-33017

[2] GitHub Security Advisory, "GHSA-vwmf-pq79-vjvx: Langflow Unauthenticated RCE via Public Flow Build Endpoint," GitHub, March 17, 2026. https://github.com/langflow-ai/langflow/security/advisories/GHSA-vwmf-pq79-vjvx

[3] Sysdig Threat Research Team, "CVE-2026-33017 — How Attackers Compromised Langflow AI Pipelines in 20 Hours," Sysdig, March 2026. https://www.sysdig.com/blog/cve-2026-33017-how-attackers-compromised-langflow-ai-pipelines-in-20-hours

[4] CISA, "CISA Adds CVE-2026-33017 to Known Exploited Vulnerabilities Catalog," CISA KEV, March 25, 2026. https://www.cisa.gov/known-exploited-vulnerabilities-catalog

[5] CISA, "CISA Adds CVE-2025-3248 to Known Exploited Vulnerabilities Catalog," CISA KEV, May 5, 2025. https://www.cisa.gov/known-exploited-vulnerabilities-catalog

[6] DataStax, "Langflow Documentation," DataStax, 2026. https://docs.datastax.com/en/langflow/index.html

[7] z4yd3, "PoC-CVE-2026-33017," GitHub, March 2026. https://github.com/z4yd3/PoC-CVE-2026-33017

[8] MaxMnMl, "langflow-CVE-2026-33017-poc," GitHub, March 2026. https://github.com/MaxMnMl/langflow-CVE-2026-33017-poc

[9] Obsidian Security, "CVE-2025-34291 — Critical Account Takeover and RCE in Langflow," Obsidian Security Blog, 2025. https://www.obsidiansecurity.com/blog/cve-2025-34291-critical-account-takeover-and-rce-vulnerability-in-the-langflow-ai-agent-workflow-platform

[10] Qualys ThreatPROTECT, "CISA Added Langflow Vulnerability CVE-2026-33017 to Its Known Exploited Vulnerabilities Catalog," Qualys, March 26, 2026. https://threatprotect.qualys.com/2026/03/26/cisa-added-langflow-vulnerability-to-its-known-exploited-vulnerabilities-catalog-cve-2026-33017/

[11] Barrack AI, "Langflow Got Hacked Twice Through the Same exec() Call," Barrack AI Blog, March 2026. https://blog.barrack.ai/langflow-exec-rce-cve-2026-33017/

[12] Horizon3.ai, "Unsafe at Any Speed — Abusing Python exec for Unauth RCE in Langflow AI (CVE-2025-3248)," Horizon3.ai Attack Research, 2025. https://horizon3.ai/attack-research/disclosures/unsafe-at-any-speed-abusing-python-exec-for-unauth-rce-in-langflow-ai/

[13] The Hacker News, "Critical Langflow Flaw CVE-2026-33017 Triggers Attacks within 20 Hours," The Hacker News, March 2026. https://thehackernews.com/2026/03/critical-langflow-flaw-cve-2026-33017.html

[14] GitLab Advisory Database, "CVE-2026-33017," GitLab, March 2026. https://advisories.gitlab.com/pkg/pypi/langflow/CVE-2026-33017/

[15] Langflow, "100K Stars on GitHub," Langflow Blog, 2024. https://www.langflow.org/blog/100k-stars-on-github