Langflow CVE-2026-33017: AI Workflow Hijacking via Unauthenticated RCE

Unofficial AI-assisted Research

Cloud Security Alliance AI Safety Initiative

2026-03-29

Key Takeaways

- CVE-2026-33017 is a critical (CVSS 9.3) unauthenticated remote code execution vulnerability in Langflow, affecting all versions prior to 1.9.0, exploitable via a single crafted HTTP request with no authentication or prior access required [1][2].
- Active exploitation began within 20 hours of public disclosure on March 17, 2026, before any public proof-of-concept existed — a timeline that reflects both the scale of automated scanning infrastructure and the capability of adversaries to develop working exploits from advisory text alone [3].
- CISA added the vulnerability to its Known Exploited Vulnerabilities (KEV) catalog on March 25, 2026, requiring Federal Civilian Executive Branch (FCEB) agencies to remediate by April 8, 2026 [4].
- Successful exploitation yields access to all AI service credentials, cloud API keys, and database connection strings stored within the compromised Langflow environment. Depending on deployment configuration, this can enable lateral movement across an organization's broader AI infrastructure [3][5].
- Organizations must upgrade to Langflow 1.9.0 immediately [1][4] and treat all credentials previously accessible to the compromised instance as potentially compromised regardless of direct evidence of exfiltration [3].

Background

Langflow is an open-source, low-code platform for building and deploying AI-powered agents and workflows. Developed by the Langflow AI project and backed in part by DataStax (which IBM announced plans to acquire in early 2025), Langflow has accumulated over 100,000 GitHub stars and seen broad adoption across enterprise agentic AI pipeline development, making it a high-value target for adversaries [6][7]. Its core design allows developers to compose workflows visually by connecting nodes representing large language models, retrieval-augmented generation (RAG) components, external tools, and custom Python logic. These flows are stored as structured JSON graphs and executed by a Python-based runtime engine that orchestrates data movement across components.

The platform's appeal in enterprise contexts derives directly from its integration breadth. A single Langflow instance commonly holds configured API credentials for providers such as OpenAI, Anthropic, and AWS, alongside database connection strings, vector store endpoints, and webhook secrets for downstream automation. This makes Langflow a high-value target: an attacker who compromises the runtime environment does not need to pivot across multiple systems. The credentials aggregated within a single deployment can provide simultaneous access to an organization's AI budget and model access, cloud storage and compute resources, customer databases, and any automated pipelines that consume those credentials [3][5].

The vulnerability CVE-2026-33017 was discovered by security researcher Aviral Srivastava, who reported the flaw on February 26, 2026. The advisory became public on March 17, 2026, following the release of Langflow version 1.9.0, which contained the fix [1][8].

Security Analysis

Root Cause: Unsandboxed Execution of Attacker-Controlled Flow Data

The vulnerability resides in the `POST /api/v1/build_public_tmp/{flow_id}/flow` endpoint, a feature designed to allow unauthenticated users to build and preview publicly-shared flows. The intended behavior is for this endpoint to retrieve a stored, validated flow definition from the database and execute it. The flaw arises from an optional `data` parameter: when supplied, the endpoint substitutes the attacker-provided flow definition in place of the stored one, and the server-side runtime passes the resulting node definitions directly to Python's `exec()` function with no sandboxing, input validation, or code-signing check [1][2][3].

The consequence is that an unauthenticated attacker who sends a single HTTP POST request containing a JSON payload with malicious Python code in a node definition achieves immediate remote code execution on the server as the user running the Langflow process. No session, no token, no credential, and no multi-step exploitation chain is required. The vulnerability is classified under CWE-94 (Improper Control of Generation of Code) and CWE-306 (Missing Authentication for Critical Function) [1][9].

This design pattern — executing flow graphs as code without a hard isolation boundary between the workflow definition layer and the server runtime — has appeared in prior visual automation frameworks, suggesting a recurring architectural assumption that warrants category-level scrutiny beyond any single product. The vulnerability is not attributable to a single line of careless code but rather to the absence of

a trust boundary between the data plane (flow definitions) and the execution plane (the Python runtime). This distinction matters for organizations assessing the security of similar platforms in their environments.

Exploitation Timeline and Observed Attack Behavior

The Sysdig Threat Research Team instrumented honeypot nodes to capture exploitation activity and documented a three-phase attack pattern following advisory publication [3].

In the first phase, beginning approximately 20 to 21 hours after disclosure, automated scanning infrastructure using the Nuclei security testing framework began probing for vulnerable instances. The characteristic reconnaissance payload executed the `id` command, base64-encoded its output, and exfiltrated the result to callback servers operated under the `.oast.live`, `.oast.me`, and `.oast.pro` domains — infrastructure associated with the open-source interactsh out-of-band interaction tool commonly used by offensive security practitioners. Sysdig attributed this phase to at least four distinct source IP addresses consistent with nuclei scanner infrastructure [3].

The second phase, running from approximately 21 to 24 hours post-disclosure, involved more sophisticated operators deploying custom Python exploitation scripts. These scripts performed methodical reconnaissance: directory listing, system fingerprinting, and enumeration of the running process environment before attempting staged payload delivery. The presence of custom tooling so early in the exploitation window, before any public proof-of-concept code existed, is consistent with actors who developed working exploits independently following advisory publication [3].

In the third and most consequential phase, observed from roughly 24 to 30 hours after disclosure, operators conducted systematic credential harvesting. Targets included `.env` files containing database connection strings and AI service API keys, `/etc/passwd` for system enumeration, SQLite database files located through directory traversal, and environment variables surfaced via the Python `os.environ` dictionary. Sysdig documented a command-and-control server at `143.110.183.86:8080` and a dropper host at `173.212.205.251:8443` associated with at least one observed intrusion chain [3].

The kill chain observed was consistent across operators: code execution via `os.popen()`, base64 encoding of output, and HTTP-based exfiltration to callback infrastructure. Detection teams can apply this behavioral signature in process monitoring rules and egress filtering policies as a CVE-agnostic indicator.

AI Workflow Hijacking as a Distinct Threat Class

CVE-2026-33017 is not simply an RCE vulnerability that happens to affect an AI product. The mechanism of exploitation — injecting malicious logic into a workflow definition that the platform then faithfully executes — represents a distinct threat class with implications beyond traditional server compromise. When an attacker inserts a malicious node into a Langflow workflow and causes it to execute, they are operating within the same execution model that the platform was designed to support. From the runtime's perspective, the malicious payload is executed identically to legitimate business logic — the same interpreter, the same process, the same permissions. Detection must therefore rely on behavioral signals at adjacent layers: process execution patterns, network egress destinations, and anomalous file access, rather than application-layer content inspection.

This characteristic makes detection and containment harder than for conventional intrusions. Network defenders who have instrumented the Langflow API for monitoring legitimate workflow traffic may not have alerting tuned to distinguish a malicious node execution from a legitimate one, especially in environments with active development workflows. The attacker's code runs with the full context of the workflow engine: it can read stored credentials, invoke configured integrations, interact with connected databases, and send requests to external services, all using the platform's existing authenticated sessions. An attacker with sustained access could modify live production flows to exfiltrate data from every query processed by the pipeline, turning the AI system itself into a persistent data exfiltration channel.

Broader Implications: AI Infrastructure as a Credential Aggregation Point

The credential harvesting behavior observed in active exploitation reflects a broader strategic reality about how AI infrastructure is deployed. Unlike a traditional application server, a Langflow deployment is an integration hub: it is provisioned with credentials for every downstream service it interacts with. Credential hygiene in these environments is frequently limited by the absence of integrated secrets management in developer tooling, with API keys stored in `.env` files or environment variables rather than external vaults [3][5]. In many deployments, database credentials are embedded in connection strings and cloud provider keys are configured at deployment time; organizations using external secrets management vaults may have a reduced credential exposure scope, though environment variable injection at runtime still presents a harvesting surface. The compromise of one Langflow instance in a multi-environment organization may therefore yield credentials spanning development, staging, and production systems, depending on how secrets are managed.

Any orchestration platform that brokers access to multiple AI services — whether MCP server deployments, LangChain agent runtimes, AutoGen multi-agent environments, or comparable frameworks — shares the credential aggregation characteristic that made Langflow a high-value target. The specific risk profile varies by execution isolation model and credential management design, but the aggregate access risk warrants comparable scrutiny. CVE-2026-33017 should be read as a category signal, not solely as a product-specific incident.

Recommendations

Immediate Actions

Organizations running any version of Langflow prior to 1.9.0 should treat the situation as an active incident until patching is complete. The upgrade to Langflow 1.9.0 or later is the primary remediation and should be prioritized for all internet-accessible deployments, followed by internal deployments [1][2][4]. If patching cannot be completed immediately, the `build_public_tmp` endpoint family should be blocked at the network perimeter or web application firewall layer until the upgrade is applied. This endpoint serves the public flow preview feature and is not required for authenticated workflow operation.

Credential rotation should occur in parallel with patching and should not wait for forensic confirmation of exfiltration. Every API key, database credential, cloud provider key, and service token accessible to the Langflow environment should be treated as potentially compromised and rotated. Organizations should coordinate with AI service providers (OpenAI, Anthropic, and others) to revoke and reissue API keys, and with cloud providers to audit access logs for the affected key material over the exploitation window from March 17 onward.

Short-Term Mitigations

Network architecture review should follow immediately after patching. Langflow instances should not be directly internet-accessible; deployments should be placed behind authentication proxies or VPN boundaries with access limited to known development and operations principals [3][5]. Egress filtering at the host or container level should restrict outbound traffic from the Langflow process to only the endpoints required for configured integrations, eliminating the base64-exfiltrate-to-callback pattern observed in active exploitation.

Detection teams should create alerting for the behavioral signatures documented during exploitation: execution of shell commands via `os.popen()` or `subprocess` from the Langflow process, outbound DNS lookups to `.oast.live`, `.oast.me`, or `.oast.pro` domains, and unusual file reads targeting `.env`, `.db`, and `/etc/passwd` paths. Runtime security tools such as Falco or comparable eBPF-based process monitoring can detect these behaviors through anomaly baselines rather than requiring CVE-specific signatures [3].

Organizations should audit the scope of credentials stored in their Langflow deployments and evaluate whether all configured integrations remain current and necessary. Removing unused credential bindings reduces the blast radius of any future exploitation event affecting the platform or its successors.

Strategic Considerations

The speed of exploitation in this incident — working attacks before any public proof-of-concept, driven by adversaries reading the same advisory text that defenders received — compresses the time available for organizations to patch after a disclosure. Vulnerability management programs that rely on extended remediation windows — whether 30, 60, or 90 days — are incompatible with the operational tempo observed here. Security teams should evaluate whether AI infrastructure components warrant an expedited remediation track analogous to the treatment given to perimeter network devices, given their role as credential aggregation points.

The design pattern that enabled CVE-2026-33017 — the absence of a trust boundary between flow definitions as data and the server runtime as executable environment — should inform procurement and security review criteria for AI orchestration platforms. Organizations evaluating Langflow or comparable tools should require documentation of the execution isolation model, including whether workflow logic is executed in a sandboxed subprocess, constrained by a policy engine, or runs with full access to the host runtime environment. For platforms where workflow execution is equivalent to arbitrary code execution, compensating controls at the infrastructure layer (containerization with restricted capabilities, service accounts with minimal permissions, secrets management through external vaults rather than environment variables) should be treated as baseline requirements rather than optional hardening, given the execution model equivalence.

CSA Resource Alignment

CVE-2026-33017 maps to several active threat categories addressed in CSA's research and guidance frameworks.

The **MAESTRO framework** for agentic AI threat modeling is directly applicable to the attack pattern observed in this incident. MAESTRO's layered threat model for multi-agent systems addresses the risk of adversarial manipulation at the orchestration layer — precisely the attack surface that CVE-2026-33017 exposes. The exploit traverses from attacker-controlled input (Layer 1: model and data inputs) through the orchestration plane (Layer 4: agent orchestration) to the underlying infrastructure (Layer 6: execution environment), exploiting the absence of a trust boundary between layers. Organizations applying MAESTRO to their Langflow deployments should evaluate trust boundaries between flow definition ingestion and runtime execution as a first-tier control assessment [10].

The **CSA Cloud Controls Matrix (CCM)** and the **AI Controls Matrix (AICM)**, which supersedes and extends the CCM for AI-specific deployments, addresses the control domains most directly violated by this vulnerability. The applicable control families include Application and Interface Security (AIS), covering secure coding and input validation; Identity and Access Management (IAM), covering authentication requirements for sensitive endpoints; and Data Security (DSI), covering the protection of credentials and connection strings in AI system configurations. Organizations conducting CCM or AICM gap assessments should verify that their AI orchestration platforms are covered under these control families with appropriate compensating controls documented [11].

The **CSA Zero Trust Guidance** applies to the network posture recommendations in this advisory. The pattern of Langflow deployments directly exposed to the internet without authentication intermediaries represents a violation of zero trust principles that assume breach and require verification for all access. Placing AI orchestration platforms inside a zero trust perimeter — requiring authenticated, authorized, and continuously validated access for all API consumers — substantially reduces the exploitability of unauthenticated endpoint vulnerabilities of which CVE-2026-33017 is an instance, by preventing unauthenticated actors from reaching the vulnerable endpoint.

The **CSA AI Organizational Responsibilities** guidance addresses the governance dimension of this incident. The rapid exploitation timeline underscores the need for AI infrastructure to be included in vulnerability management programs with the same rigor applied to production application servers, and for AI system owners to maintain current inventories of all credentials and integrations configured in their deployed platforms. Without this inventory, the credential rotation step required after this incident cannot be executed completely.

References

[1] CVE-2026-33017 — CVE Details. "CVE-2026-33017: Langflow Unauthenticated RCE via Public Flow Build Endpoint." CVEDetails.com. March 2026. https://www.cvedetails.com/cve/CVE-2026-33017/ *(Note: page may return 403; see also NVD entry at https://nvd.nist.gov/vuln/detail/CVE-2026-33017)*

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

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

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

[5] BleepingComputer. "CISA: New Langflow Flaw Actively Exploited to Hijack AI Workflows." BleepingComputer.com. March 2026. https://www.bleepingcomputer.com/news/security/cisa-new-langflow-flaw-actively-exploited-to-hijack-ai-workflows/

[6] Langflow AI. "Langflow: Low-Code AI Builder for Agentic and RAG Applications." Langflow.org. 2026. https://www.langflow.org/

[7] GitHub — langflow-ai/langflow. "Langflow is a powerful tool for building and deploying AI-powered agents and workflows." GitHub.com. 2026. https://github.com/langflow-ai/langflow

[8] SecPod Blog. "CVE-2026-33017: Critical Langflow Vulnerability Exploited Within 20 Hours of Disclosure." SecPod.com. March 2026. https://www.secpod.com/blog/cve-2026-33017-critical-langflow-vulnerability-exploited-within-20-hours-of-disclosure/

[9] Help Net Security. "CISA Sounds Alarm on Langflow RCE, Trivy Supply Chain Compromise After Rapid Exploitation." HelpNetSecurity.com. March 27, 2026. https://www.helpnetsecurity.com/2026/03/27/cve-2026-33017-cve-2026-33634-exploited/

[10] Cloud Security Alliance. "MAESTRO: Multi-Agent Environment for Systematic Threat and Risk Operations." CSA AI Safety Initiative. https://cloudsecurityalliance.org/

[11] Cloud Security Alliance. "AI Controls Matrix (AICM)." CSA Research. https://cloudsecurityalliance.org/