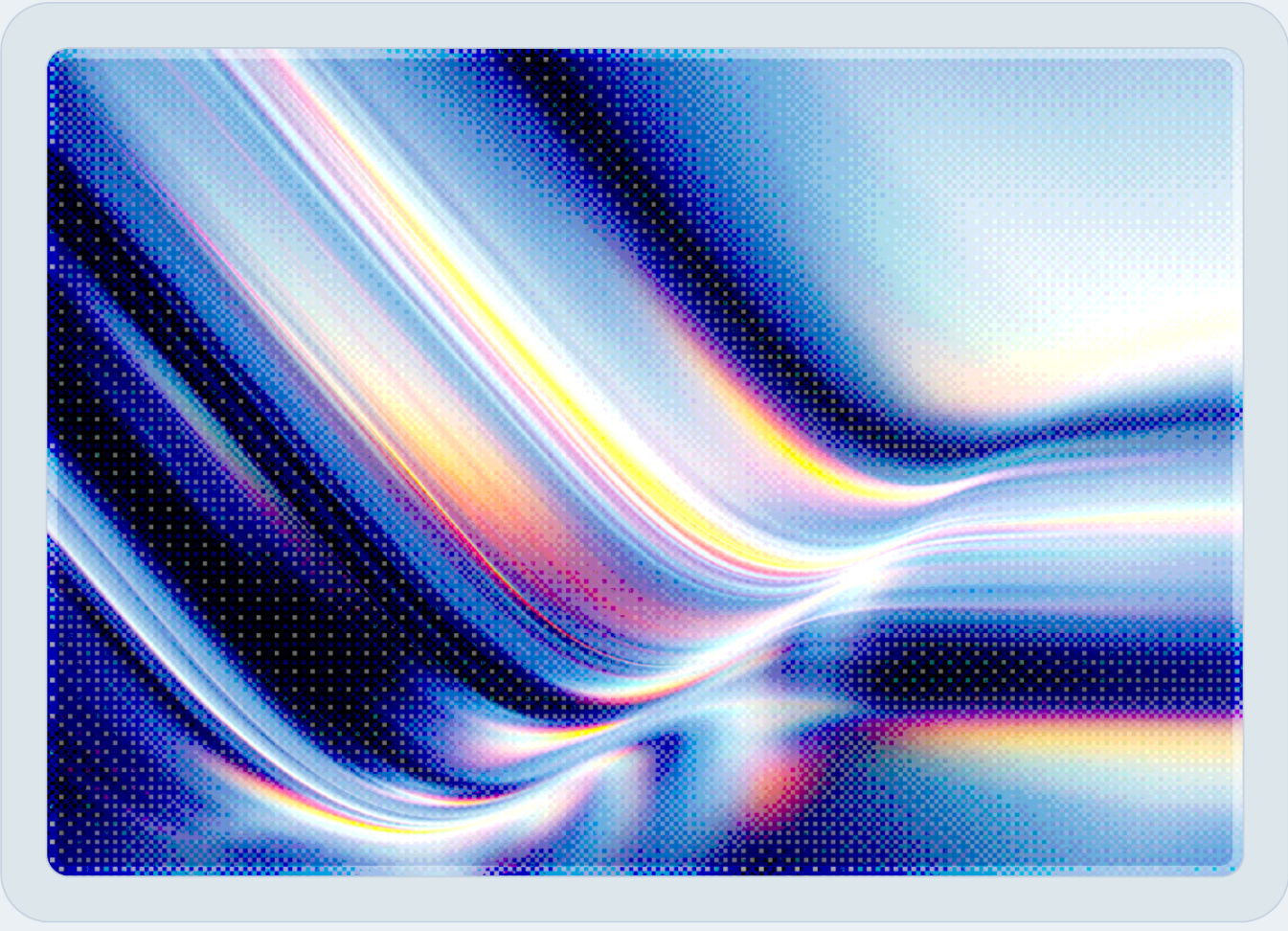
CSAI Foundation | Cloud Security Alliance

NATS-as-C2: AI Pipeline RCE Fuels Credential Harvesting Campaign

How Langflow CVE-2026-33017 Weaponized Cloud Messaging to Steal AI API Keys

2026-05-15

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

- A threat actor exploited CVE-2026-33017, an unauthenticated remote code execution flaw in the Langflow AI workflow platform, to extract AWS credentials and AI provider API keys directly from container environment variables within 20 hours of the vulnerability's public disclosure [1][2].
- The attacker deployed a distributed worker toolkit called "KeyHunter" that used a legitimate NATS messaging server as its command-and-control backbone — the first documented use of NATS-as-C2 infrastructure in a credential harvesting campaign [1].
- The KeyHunter worker validates AWS access keys, LLM provider API keys (OpenAI, Anthropic, Google), and scans cloud development environments for exposed secrets, enabling rapid monetization through LLMjacking and unauthorized cloud compute abuse [1][3].
- CISA added CVE-2026-33017 to its Known Exploited Vulnerabilities catalog on March 25, 2026, with a Federal Civilian Executive Branch remediation deadline of April 8, 2026; organizations running Langflow 1.8.1 or earlier should treat this as a critical priority [4][5][13].
- The incident illustrates a maturing attacker playbook in which AI infrastructure vulnerabilities serve as the entry point for credential-focused operations targeting the rapidly growing financial value of LLM API access [1][6].

Background

Langflow is an open-source, visual framework for building AI agent pipelines and retrieval-augmented generation (RAG) applications. Released by DataStax and widely adopted in the AI developer community for rapid prototyping and production deployment of AI workflows, its low-code drag-and-drop interface connects large language models, data sources, and tool integrations with minimal boilerplate. That accessibility has made Langflow popular among organizations standing up AI infrastructure quickly — and that same popularity has made internet-facing Langflow deployments a high-value target for threat actors.

On March 17, 2026, a critical unauthenticated remote code execution vulnerability – CVE-2026-33017 – was disclosed in Langflow versions up to and including 1.8.1 [2]. The flaw resides in the `/api/v1/build_public_tmp/{flow_id}/flow` endpoint, which is designed to allow public preview of workflow flows. The endpoint accepts attacker-supplied flow data containing arbitrary Python code in node definitions and executes that code server-side without any sandboxing or authentication requirement [2][3]. A single unauthenticated HTTP POST request is sufficient to achieve full remote code execution on any exposed Langflow instance. The advisory description provided sufficient technical detail that attackers appear to have been able to construct working exploits without additional research, as evidenced by exploitation beginning within 20 hours of disclosure [2].

Langflow has experienced a pattern of critical RCE vulnerabilities that warrants broader notice. CVE-2025-3248, an unauthenticated RCE in versions prior to 1.3.0 that was added to the CISA KEV catalog in May 2025, was actively exploited to deploy the Flodrix botnet [12]. CVE-2025-34291 (CVSS 9.4), disclosed in early 2026, enables account takeover and RCE via a CORS-based attack requiring only that a victim visit a malicious webpage [8]. Each of these vulnerabilities was exploited in the wild shortly after disclosure. The pattern establishes Langflow as a consistently targeted platform, and organizations cannot rely on infrequent patching cadences for internet-facing Langflow instances.

NATS – a Cloud Native Computing Foundation (CNCF) project – provides high-performance, lightweight publish/subscribe messaging for distributed systems and microservices [9]. Its JetStream extension adds durable message persistence, stream replay, and consumer acknowledgement semantics, enabling fault-tolerant event-driven architectures [10]. NATS is widely deployed in cloud-native environments and, crucially, its traffic profile can be difficult to distinguish from legitimate application messaging in environments that do not maintain egress inventories or NATS-aware connection baselines – a property the KeyHunter operator appears to have deliberately exploited when selecting it as a C2 channel.

Security Analysis

The Sysdig Threat Research Team documented this campaign based on honeypot data and threat intelligence gathered following CVE-2026-33017's disclosure, publishing what appears to be the first public analysis of a NATS server used as command-and-control infrastructure for a credential harvesting operation [1]. The attack is notable both for the speed of its weaponization and for the architectural sophistication of the post-exploitation tooling.

The Attack Chain

The intrusion unfolded over approximately 10 hours — from initial probing at 04:13 UTC to KeyHunter deployment at 14:03 UTC — with the first exploitation of CVE-2026-33017 occurring within 20 hours of the vulnerability's public disclosure [1][2]. Initial probes targeting LMDeploy and LiteLLM instances began at 04:13 UTC, with reconnaissance shifting to Langflow by 09:09 UTC. Within three minutes of the first Langflow probe, the attacker achieved RCE via CVE-2026-33017, executing a payload that captured environment variables including `AWS_ACCESS_KEY_ID` and `AWS_SECRET_ACCESS_KEY` [1][2]. The attacker validated those credentials by calling `sts:GetCallerIdentity`, then conducted a systematic reconnaissance sweep across S3, EC2, Lambda, CloudWatch, ECS, SageMaker, and IAM services. By 14:03 UTC, the attacker had deployed the KeyHunter worker binary to the compromised host, establishing persistent connectivity to a NATS C2 server at 45.192.109.25:14222 [1].

The exploitation technique was direct. By injecting malicious Python payloads via JSON into flow node definitions, the attacker executed shell commands through Python's `os.popen()` and exfiltrated results via HTTP callback to attacker-controlled staging infrastructure. The progression from an initial validation probe (executing `id`) to systematic harvesting — running `env` to capture all environment variables, then `find /app -name "*.db" -o -name "*.env"` to locate additional secrets on disk — followed a methodical, automated sequence consistent with a prepared toolkit [2].

The KeyHunter Worker Architecture

The post-exploitation payload deployed to compromised hosts reflects substantial engineering investment, distinguishing this operation from opportunistic script-based exploitation. The operator named the project "KeyHunter" [1] and delivered it in two formats: a 9.4 MB Go binary (`worker-linux-amd64`) built for both x86_64 and aarch64 architectures, and a Python fallback script (`keyhunter_worker.py`) for environments where the Go binary cannot execute. The Go binary's module path (`github.com/keyhunter/worker`) and a leaked Windows project path (`/AyuGram Desktop/KeyHunter-Distributed/worker/`) suggest active development with cross-platform ambitions [1].

Once installed, the worker connects to the attacker's NATS server over an authenticated, ACL-enforced session and subscribes to four task subjects representing distinct monetization strategies. The `task.validate_aws` subject triggers AWS key validation through boto3's `sts:GetCallerIdentity` call; `task.validate_ai` validates LLM provider credentials against vendor authentication endpoints, covering OpenAI, Anthropic, Google, GitHub, Slack, and Stripe; `task.scan_cde` targets cloud development environments — CodePen, JSFiddle, StackBlitz, and CodeSandbox — to harvest API keys embedded in shared code snippets; and `task.scan_web`

enables arbitrary URL scraping against attacker-specified targets [1]. The credential capture subsystem embeds a 12-pattern regex set covering AWS access keys, bearer tokens, JWTs, private keys, and database connection strings, enabling automated classification and triage of harvested material.

Workers publish results to the NATS server via enumerated response subjects, using JetStream durable pull consumers with explicit acknowledgements to guarantee reliable delivery even across network interruptions. A `heartbeat.worker` subject provides the operator with a live view of which infected hosts remain active. The deployment script installs the worker as a systemd service configured with `Restart=always` and `RestartSec=5`, ensuring persistence across reboots and process crashes [1]. The worker also integrates uTLS fingerprint mimicry to defeat bot-detection controls deployed by the code-sharing platforms it targets — a detail indicating that the tooling was iteratively refined against real defenses.

Why NATS as C2

The selection of NATS as command-and-control infrastructure appears architecturally deliberate. In cloud-native environments, outbound NATS connections are operationally routine; containerized microservices regularly communicate over NATS, and in environments lacking NATS-aware behavioral detection, that traffic is unlikely to trigger signature-based controls tuned for traditional botnet protocols or HTTP-based C2 channels. By deploying an authenticated, ACL-enforced NATS instance rather than a conventional malware C2 framework, the attacker created a resilient, multi-worker coordination plane that blends into the expected network behavior of the environments being targeted [1].

JetStream's durable consumer semantics provide a resilience property uncommon in traditional C2 architectures: tasks persist at the broker until an individual worker explicitly acknowledges completion, naturally tolerating worker churn without operator intervention. Workers installed on Langflow containers that are subsequently patched or restarted will receive and execute queued tasks upon reconnection. This architecture tolerates worker churn more gracefully than typical polling-based C2 frameworks, providing operational resilience against partial remediation efforts.

Financial Stakes: The LLMjacking Economy

The credential types targeted by KeyHunter directly reflect the escalating financial value of AI API access. According to Prompt Guardrails, stolen LLM credentials sell for approximately $30 per month of access on criminal markets [6], while the underlying compute charges from LLMjacking can far exceed this value: Sysdig's analysis of LLMjacking economics calculates that unauthorized use of AWS Bedrock Claude 2.x at compute quota limits can generate over $46,000 per day in charges for the victim [14].

Vendor-reported case studies document individual LLMjacking incidents generating charges sufficient to force business interruptions within 48-hour windows [6]. These figures represent direct financial exposure from a single set of stolen credentials, independent of any reputational or operational harm.

The supply of stolen credentials feeding this market is substantial and growing. GitGuardian's analysis found nearly 29 million new secrets exposed in public GitHub commits across 2025 — a 34% year-over-year increase [7]. Sysdig's threat research documented a 376% increase in credential theft specifically targeting AI services between Q4 2025 and Q1 2026 [1]. The KeyHunter operation's deliberate targeting of AI API keys alongside cloud infrastructure credentials reflects a mature threat actor understanding that LLM access is now a monetizable asset comparable to raw compute.

Recommendations

Immediate Actions

Organizations running Langflow must upgrade to version 1.9.0 or later as the only complete remediation for CVE-2026-33017. Instances that cannot be immediately upgraded should be taken offline or placed behind a WAF with rules blocking requests to `/api/v1/build_public_tmp/`. Any Langflow deployment that was internet-facing before the patch was applied should be treated as potentially compromised: rotate all credentials accessible from the container environment, including AWS access keys and any AI provider API keys, without waiting to confirm breach indicators. The combination of 20-hour time-to-exploitation and automated environment variable harvesting means that exposure duration alone is not a reliable indicator of whether credentials were stolen.

Security teams should audit environment variable usage across all AI pipeline components as an urgent follow-on action. AWS credentials, LLM API keys, and other secrets injected into containerized workloads as environment variables are fully accessible to any attacker who achieves RCE within that container. Secrets should instead be delivered through dedicated secrets management systems — AWS Secrets Manager, HashiCorp Vault, or Kubernetes Secrets with envelope encryption — with short-lived credentials and just-in-time delivery wherever operationally feasible.

Short-Term Mitigations

Runtime security monitoring should be deployed on AI infrastructure containers with detection rules targeting the specific behaviors observed in this campaign: environment enumeration commands (`env`, `printenv`), file discovery operations targeting `.env` and database files, outbound NATS

connections to non-inventory IP addresses, unexpected systemd service creation, and DNS queries to OAST callback domains such as `.oast.live` [2]. Falco-compatible rules covering sensitive file reads and curl-based stage-two payload delivery are a practical baseline that covers the documented attack chain.

Network egress controls for AI workloads deserve immediate review. AI pipeline components should operate with allowlisted egress paths limited to known vendor API endpoints and data sources. Outbound connections to IP addresses not in the service's expected communication profile — particularly on port 4222 (the NATS default) or 14222 (as used in this campaign's C2 infrastructure) — should be blocked by default and generate security alerts. This is particularly important in containerized environments where NATS-based communication might otherwise appear legitimate.

IAM policies governing credentials used by AI workloads should be audited against least-privilege principles and scoped to only the services the pipeline requires. AWS CloudTrail should be enabled across all regions with alerting configured for anomalous API call sequences, particularly enumeration patterns spanning multiple services from unusual source addresses or occurring outside normal operational windows. Enabling AWS Bedrock usage alerts and setting billing anomaly thresholds provides an additional detection layer for LLMjacking activity that bypasses infrastructure-level monitoring.

Strategic Considerations

The NATS-as-C2 technique — though documented here for the first time — fits a broader observed trend in which threat actors evaluate cloud-native tooling for C2 use precisely because it blends into legitimate operational traffic. Security operations centers should update threat hunt playbooks to include cloud-native messaging protocols — NATS, MQTT, Apache Kafka, and AMQP-based systems — as potential C2 channels rather than treating them exclusively as legitimate infrastructure. Behavioral anomaly detection that flags unexpected outbound connections to messaging brokers, regardless of protocol, is more durable than blocklists tied to specific known-bad addresses.

Organizations deploying AI pipeline infrastructure should integrate security review gates into their AI development lifecycle. Langflow, LiteLLM, and similar frameworks occupy an unusually privileged position in enterprise environments: they are designed to execute arbitrary code, they hold credentials for expensive cloud and AI services, and in many organizations they are deployed by teams whose primary focus is model development — teams that may not have deep familiarity with infrastructure hardening practices. These platforms require the same hardening discipline applied to other high-privilege workloads — regular vulnerability scanning, network segmentation from production systems, and runtime behavioral monitoring — applied before they reach production rather than after.

AI API key lifecycle management should be elevated to a first-class security control. Keys should carry short expiration windows, per-key rate limits, and model-access restrictions aligned with the specific workload using them. Several major AI providers, including OpenAI and Google, offer usage-based alerting mechanisms; when properly configured, these can surface anomalous consumption patterns early in a LLMjacking campaign. Treating API key management as an access management problem — with the same rigor applied to privileged human accounts — represents a high-impact mitigation for the monetization pathway that makes operations like KeyHunter financially viable.

CSA Resource Alignment

This incident maps to multiple layers of the CSA MAESTRO threat modeling framework for agentic AI systems [11]. At the Deployment Infrastructure layer, CVE-2026-33017 represents an authentication control failure in publicly exposed AI workflow infrastructure. At the Agent Frameworks layer, the attacker's ability to inject arbitrary Python code through flow node definitions reflects the inherent risk of trusting user-supplied content within agentic execution environments that process Python by design. The NATS-based C2 architecture demonstrates how the communications fabric of the agent ecosystem can be weaponized when outbound connectivity from AI workloads is not constrained to expected destinations.

The CSA AI Controls Matrix (AICM) provides structured control guidance relevant to organizations seeking to systematically address risks of this nature [15]. AICM controls applicable to AI Application Providers and Cloud Service Providers include secrets management, container workload hardening, network segmentation, and runtime behavioral monitoring — each directly applicable to Langflow deployments and AI pipeline infrastructure more broadly. Organizations seeking a structured assessment of their AI pipeline security posture can apply the AICM as an audit framework to identify gaps in credential handling and monitoring coverage.

CSA's Zero Trust working group guidance addresses the lateral movement dimension of this campaign [16]. The breadth of reconnaissance and service enumeration the attacker conducted using harvested AWS credentials — spanning S3, EC2, Lambda, SageMaker, IAM, and more — illustrates the blast radius that Zero Trust architecture is designed to contain. Workload-level micro-segmentation, identity-based access controls, and just-in-time credential provisioning for AI pipeline components would meaningfully limit the damage any single compromised Langflow container could enable. The principle that no single compromised workload should yield credentials with broad control-plane access is directly applicable here.

CSA's published research on AI Organizational Responsibilities and the Agentic AI Red Teaming Guide provide complementary governance and adversarial testing frameworks for organizations seeking to validate AI infrastructure security posture beyond technical controls.

References

[1] Sysdig Threat Research Team. "NATS-as-C2: Inside a new technique attackers are using to harvest cloud credentials and AI API keys." Sysdig Blog, 2026.

[2] Sysdig Threat Research Team. "CVE-2026-33017: How attackers compromised Langflow AI pipelines in 20 hours." Sysdig Blog, March 2026.

[3] CybersecurityNews. "Langflow CVE-2026-33017 Exploited to Steal AWS Keys and Deploy NATS Worker." CybersecurityNews, March 2026.

[4] CISA. "Known Exploited Vulnerabilities Catalog — CVE-2026-33017." CISA, March 25, 2026.

[5] Help Net Security. "CISA sounds alarm on Langflow RCE, Trivy supply chain compromise after rapid exploitation." Help Net Security, March 27, 2026.

[6] Prompt Guardrails. "LLMjacking: The $100K-Per-Day Attack Draining Enterprise AI Budgets." Prompt Guardrails Blog, 2026.

[7] Help Net Security. "29 million leaked secrets in 2025: Why AI agents credentials are out of control." Help Net Security, April 14, 2026.

[8] Obsidian Security. "CVE-2025-34291: Critical Account Takeover and RCE Vulnerability in the Langflow AI Agent & Workflow Platform." Obsidian Security, 2026.

[9] CNCF. "NATS Project." Cloud Native Computing Foundation, 2026.

[10] NATS Documentation. "JetStream." NATS Docs, 2026.

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

[12] Trend Micro. "Critical Langflow Vulnerability (CVE-2025-3248) Actively Exploited to Deliver Flodrix Botnet." Trend Micro, 2025.

[13] Qualys ThreatPROTECT. "CISA Added Langflow Vulnerability to its Known Exploited Vulnerabilities Catalog (CVE-2026-33017)." Qualys, March 26, 2026.

[14] Sysdig Threat Research Team. "LLMjacking: Stolen Cloud Credentials Used in New AI Attack." Sysdig Blog, 2024.

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

[16] Cloud Security Alliance. "Zero Trust Working Group." Cloud Security Alliance, 2025.