

AI Security Incident Case: JadePuffer Ransomware Leverages AI Agent to Automate Attacks

By NSFOCUS

Published: 2026-07-07 · Archived: 2026-07-28 02:00:58 UTC

In early July 2026, security firm Sysdig publicly disclosed a new type of ransomware attack. After gaining initial access by exploiting a Langflow vulnerability (CVE-2025-3248), the JadePuffer ransomware leveraged a Large Language Model (LLM) agent to automatically execute the entire attack chain. From reconnaissance, credential theft, lateral movement, and privilege escalation to the final file encryption, the entire process was completed autonomously by the AI agent. This marks the first documented case of a fully AI-agent-driven, end-to-end ransomware attack, signaling a new stage of autonomous and intelligent ransomware operations.

Event Background

JadePuffer is an attack operation disclosed by the Sysdig Threat Research Team in July 2026, described as the first documented case of “agentic ransomware.” Unlike traditional ransomware attacks that require manual operations by skilled threat actors, Sysdig assessed that this campaign was executed by an LLM-driven AI agent.

The attack’s entry point was an internet-exposed Langflow instance. Langflow is an open-source framework used to build AI applications and agentic workflows. Attackers exploited CVE-2025-3248 to achieve initial access. According to the National Vulnerability Database (NVD), versions of Langflow prior to 1.3.0 suffer from a code injection flaw in the /api/v1/validate/code endpoint. An unauthenticated remote attacker could construct requests to execute arbitrary code. The flaw carries a critical CVSS 3.1 score of 9.8.

CVE-2025-3248 Detail

Description
Langflow versions prior to 1.3.0 are susceptible to code injection in the /api/v1/validate/code endpoint. A remote and unauthenticated attacker can send crafted HTTP requests to execute arbitrary code.

Metrics CVSS Version 4.0 CVSS Version 3.x CVSS Version 2.0
NVD enrichment efforts reference publicly available information to associate vector strings. CVSS information contributed by other sources is also displayed.

CVSS 3.x Severity and Vector Strings:
CNA: VulnCheck Base Score: 9.8 CRITICAL Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

References to Advisories, Solutions, and Tools
By selecting these links, you will be leaving NIST webspace. We have provided these links to other web sites because they may have information that would be of interest to you. No inferences should be drawn on account of other sites being referenced, or not, from this page. There may be other web sites that are more appropriate for your purpose. NIST does not necessarily endorse the views expressed, or concur with the facts presented on these sites. Further, NIST does not endorse any commercial products that may be mentioned on these sites. Please address comments about this page to nvd@nist.gov.

URL	Source(s)	Tag(s)
https://github.com/langflow-ai/langflow/pull/6911	VulnCheck	Patch
https://github.com/langflow-ai/langflow/releases/tag/1.3.0	VulnCheck	Release Notes
https://www.cisa.gov/known-exploited-vulnerabilities-catalog?field_cve=CVE-2025-3248	CISA ADP	US Government Resource
https://www.horizon3.ai/attack-research/disclosures/unsafe-at-any-speed-abusing-python-exec-for-unauth-rce-in-langflow-ai/	VulnCheck	C-plot Third Party Advisory
https://www.vulncheck.com/advisories/langflow-unauthenticated-rce	VulnCheck	Third Party Advisory

MOIN Team
CN-SEC 中文网

The vendor patched the vulnerability on April 1, 2025, and CISA added it to its Known Exploited Vulnerabilities (KEV) catalog in early May of the same year. However, Langflow instances remain widely exposed on the internet, often deployed with minimal hardening configurations. These environments frequently contain cloud credentials and API keys, making them highly attractive targets for initial access.

Attack Mechanism Analysis

The JadePuffer attack consists of two distinct phases: first, compromising the Langflow instance that provides initial access, and second, moving laterally to the ultimate target—an independent production database server. All payloads were delivered through the Langflow RCE endpoint as Base64-encoded Python code.

Phase 1: The Langflow Instance

Upon gaining code execution privileges, the LLM immediately initiated reconnaissance: enumerating host information (system identification, kernel version, hostname, network interfaces, and running processes) while concurrently scanning environment variables and configuration files. It searched for LLM provider API keys (OpenAI, Anthropic, DeepSeek, etc.), cloud credentials (covering Alibaba Cloud, Tencent Cloud, Huawei Cloud, etc.), cryptocurrency wallets, and database credentials.

The AI agent then dumped Langflow's PostgreSQL database, stealing the credentials and API keys stored within. It then scanned the internal address space, probing various services using default credentials. During a MinIO object storage enumeration, the agent used default credentials (minioadmin:minioadmin) to list all buckets and selectively exfiltrate sensitive files like credentials.json and .env. When an initial request received an XML response instead of the expected JSON, the agent immediately adjusted its parsing logic, demonstrating its adaptive capabilities. Finally, the AI agent installed a crontab entry on the Langflow server to send a heartbeat signal back to the attacker's infrastructure every 30 minutes to maintain persistence.

Phase 2: The Production Database Server

Captured artifacts show that the true target of the attack was a publicly exposed production server running a MySQL database and the Alibaba Nacos configuration service. Nacos' authentication system suffers from a known bypass vulnerability (CVE-2021-29441), and its default JWT signing key has been public knowledge since 2020. Because this key remains unchanged in many deployments, token forging is trivial.

The AI agent connected to MySQL using root credentials (the source of which remains unknown) and then simultaneously attacked Nacos via multiple vectors: exploiting the CVE-2021-29441 authentication bypass, forging JWTs using the public default signing key, and directly injecting a backdoor admin account into the backend database via its root database access. The backdoor injection process showcased distinct autonomous self-correction capabilities—when the login attempt using the account generated by the initial payload failed, the agent automatically diagnosed a subprocess PATH issue and switched to directly importing the bcrypt library to regenerate the hash and log in successfully, all within 31 seconds and without human intervention.

After gaining control of Nacos, the AI agent encrypted all 1,342 Nacos service configuration items using MySQL's AES_ENCRYPT() function, deleted the original configuration and history tables, and created a ransom table named README_RANSOM. This table contained the ransom demands, a Bitcoin payment address, and a Proton Mail contact address. The AES key was generated by concatenating random UUIDs and was only printed to standard output once, never being persisted or transmitted; this means that even if the victim paid the ransom, the encrypted data could not be recovered. Subsequently, the LLM escalated from row-level deletion to wiping the entire database schema, self-narrating its rationale for target selection.

Characteristics Analysis of the AI Agent Attack

Sysdig concluded that this ransomware attack was driven by an LLM based on four independent lines of evidence:

1. **Natural Language Comments:** The attack payloads contained a large volume of natural language comments explaining the objective, priority, and handling logic of each step.
2. **Real-Time Adaptive Capabilities:** During the attack, the process adjusted its parameters in real time based on failed outcomes. A prime example occurred when the Nacos login failed; within 31 seconds, the attack workflow diagnosed the root cause, deleted the broken account, regenerated the password hash, and rebuilt the admin account.

Timeline (UTC)	Activity
19:34:24	Generates a bcrypt hash via a subprocess.run(...) call and inserts the user xadmin. If bcrypt is not in the subprocess's PATH, r.stdout is empty, leaving the stored password hash as an empty string.
19:34:36	Attempts to log in using xadmin:Xploit@2026#. The attempt fails, and no token is returned.
19:34:48	12 seconds later, the agent tests two areas simultaneously: trying the default Nacos credentials (nacos:nacos) and generating a new hash via a subprocess for a simpler password (admin123), concurrently diagnosing two potential failure reasons.
19:35:07	19 seconds later, the decisive corrective payload is issued: switching from subprocess.run(...) to directly importing bcrypt, printing bcrypt.__version__ to confirm it is importable, deleting the corrupted xadmin row, and re-inserting it with the correctly generated hash.
19:35:18	Login successful.

Source: Sysdig

3. **Understanding Natural Language Context:** During operations, the LLM parsed free-text contexts presented by the target and acted upon them. This behavior only makes sense if the text was actually read and understood, rather than processed via a scanner's pattern matching.
4. **Suspicious Bitcoin Address:** The Bitcoin address in the ransom note is a standard example address from Bitcoin developer documentation, widely present in LLM training corpora. However, blockchain data shows that the address is also an active wallet, having historically received approximately 46 BTC, with the balance being transferred out immediately.

Throughout the entire operation, Sysdig captured over 600 payloads with clear objectives. Combined with the above evidence, Sysdig determined that the attack was driven by an autonomous AI agent rather than a human operator or a static toolkit.

Security Implications

This case demonstrates that ransomware is no longer the exclusive tool of highly skilled individuals. An LLM agent can chain together reconnaissance, credential theft, lateral movement, persistence, and destruction without the operator needing deep expertise in any single step.

At the same time, this case warns us that legacy vulnerabilities are being actively exploited through automation. This attack relied on issues from years ago (the 2021 Nacos authentication bypass and unchanged default signing keys) and targeted neglected, internet-exposed infrastructure. AI agents make “spray-and-pray” attacks across an entire catalog of historical vulnerabilities virtually costless, meaning the long-tail exposure of unpatched systems will only increase.

Case Summary

The JadePuffer case serves as a warning sign, marking the evolutionary direction of ransomware attack techniques. None of the individual techniques used in this case were novel or complex, but the AI model chained them together into a complete ransomware attack targeting neglected, internet-exposed infrastructure. Although the vendor patched the vulnerability, Langflow remains widely exposed on the internet and is frequently deployed with minimal hardening configurations. These environments often contain cloud credentials and API keys, making them highly attractive entry points for attacks.

Defenders must recognize that adversaries may no longer be just human attackers, but also AI agents capable of working 24/7, self-correcting in real time, and possessing contextual understanding. Security architectures must upgrade from “responding to known attack patterns” to “defending against autonomous reasoning and adaptive attacks,” incorporating AI agent behaviors into threat models to build an effective line of defense in the age of agentic attacks.

References

- [1] BleepingComputer: JadePuffer ransomware used AI agent to automate entire attack — <https://www.bleepingcomputer.com/news/security/jadepuffer-ransomware-used-ai-agent-to-automate-entire-attack/>
- [2] Sysdig: JadePuffer — Agentic Ransomware for Automated Database Extortion — <https://www.sysdig.com/blog/jadepuffer-agentic-ransomware-for-automated-database-extortion>
- [3] <https://cn-sec.com/archives/5321771.html>

Post Views: 727

Source: <https://nsfocusglobal.com/ai-security-incident-jadepuffer-ransomware-leverages-ai-agent-to-automate-attacks/>