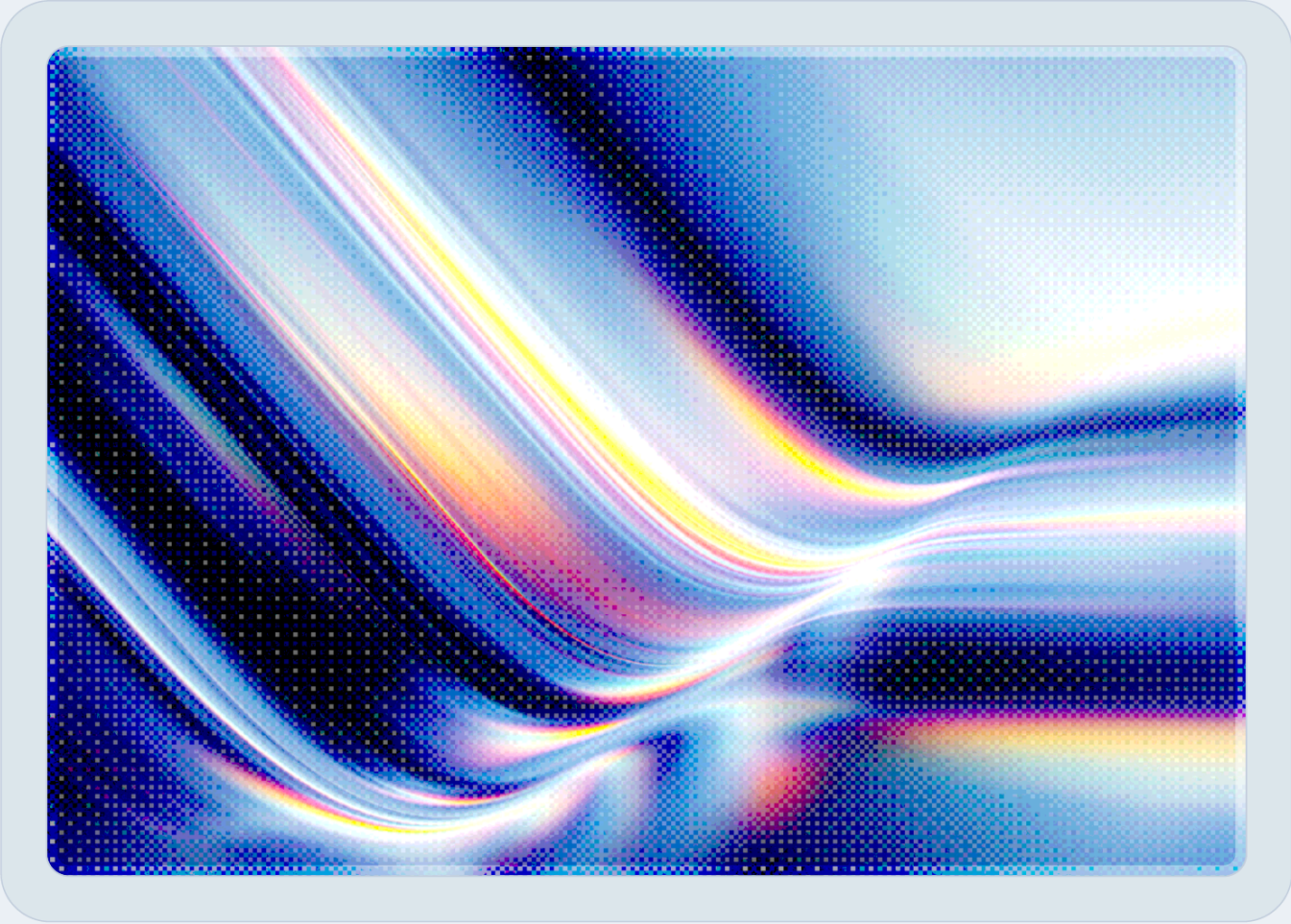
CSAI Foundation | Cloud Security Alliance

ChromaToast: Pre-Auth RCE in ChromaDB and AI Infrastructure Risk

2026-05-20

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

On May 19, 2026, HiddenLayer publicly disclosed CVE-2026-45829, a pre-authentication remote code execution flaw in the ChromaDB Python FastAPI server known as "ChromaToast." The flaw carries a CVSS v4.0 base score of 10.0 and is mapped to CWE-94, Improper Control of Generation of Code [1][2]. The vulnerability has been present since ChromaDB 1.0.0 and remained unpatched in the latest release at the time of disclosure (1.5.9). HiddenLayer's scan of internet-exposed ChromaDB instances found that approximately 73 percent are running affected versions of the Python server [1][3].

ChromaDB, with roughly 14 million monthly PyPI downloads, is among the most frequently downloaded open-source vector databases used in RAG workloads, and its customer roster includes Mintlify, Factory AI, and Weights & Biases [3][4]. The vulnerability is triggered through a single anonymous HTTP request to the collection-creation endpoint, supplying a HuggingFace model reference with the `trust_remote_code` flag set. The Python server instantiates the embedding function — and downloads and executes attacker-controlled code from the referenced HuggingFace repository — before it ever checks whether the requester is authenticated. Successful exploitation yields full process control, access to environment variables, API keys, mounted secrets, and the contents of any storage reachable from the host [1][2][5].

ChromaToast is not an isolated incident. It is the latest in a sequence of critical vulnerabilities in the open-source infrastructure that supports modern AI applications, including the "Bleeding Llama" unauthenticated memory disclosure in Ollama (CVE-2026-7482), a cluster of path-traversal, deserialization, and SQL-injection flaws disclosed in LangChain and LangGraph in March 2026, and an ongoing pattern of remote code execution defects in MLflow [6][7][8][9]. Together these incidents describe an emerging attack surface — the AI application infrastructure tier — that sits beneath the model and above the cloud, developed rapidly by small teams operating under significant adoption pressure, with limited security engineering resources, and now visibly under-secured. Security teams whose AI strategy depends on RAG, model serving, or agentic orchestration should treat this tier as a first-class system in their asset inventory, threat model, and vulnerability program.

Background

The ChromaToast Vulnerability

ChromaDB is an open-source search infrastructure project maintained by Chroma, a San Francisco company that raised an $18 million seed round in early 2024 with backing from Quiet Capital and a roster of founders and operators from Vercel, Notion, Replit, and Motherduck [10]. The project ships two server implementations: a Rust frontend invoked as `chroma run`, which the maintainers position as the default, and a Python FastAPI server reachable through the `chromadb.server.fastapi.FastAPI` entry point. The Python server remains in wide use across self-hosted deployments, development environments, and any installation that pulled the `chromadb[server]` Python distribution. Only the Python server is affected by CVE-2026-45829 [1][3].

The flaw was first reported in November 2025 by an independent researcher operating under the handle Azraelxuemo, who received no acknowledgement from the maintainers. HiddenLayer's research team identified the same defect independently and began its own disclosure attempts on February 17, 2026, escalating through direct contact, social media, and finally the IT-ISAC over the following two months without receiving substantive engagement. HiddenLayer published the technical writeup, "ChromaToast Served Pre-Auth," on May 13, 2026, and a coordinated public disclosure with the assigned CVE followed on May 19 [1][3][5]. ChromaDB version 1.5.9 had been released roughly two weeks earlier but did not address the vulnerability, and as of this writing no patched version exists [3].

Why ChromaDB Matters

Vector databases occupy a structurally central position in the AI application stack. They store the embedded representations of an organization's proprietary documents, support tickets, codebases, and internal communications — the corpus that RAG pipelines retrieve from to ground large language model outputs. A vector database compromise yields two distinct prizes: the embeddings, which research has shown can in some configurations be partially inverted to reconstruct portions of the source text, and the credentials the database process holds to communicate with embedding providers, object storage, and orchestration layers. CSA's *Securing LLM Backed Systems* guidance identified vector database security as a discrete concern in 2024 and called out the integrity of the retrieval substrate as foundational to any RAG deployment [11].

ChromaDB's developer-first ergonomics — single-binary install, no required authentication, default-open ports — reflect design choices optimized for prototype builders and rapid iteration. When an instance graduates to production without commensurate hardening, public-internet exposure is common. HiddenLayer's enumeration concluded that 73 percent of reachable ChromaDB instances were running affected versions of the Python server, consistent with a population deployed quickly and never re-hardened [1][3].

Security Analysis

How the Attack Works

The vulnerability resides in `chromadb/server/fastapi/__init__.py`, in the path handling for `POST /api/v2/tenants/{tenant}/databases/{database}/collections`. The server's handler parses the incoming JSON body and invokes `load_create_collection_configuration_from_json()`, which instantiates the embedding function specified by the client before calling `sync_auth_request()` to check whether the requester is permitted to create the collection. The attack lives in the gap between those two operations [2][12].

When a client supplies an embedding function configuration that references a HuggingFace model and includes `trust_remote_code: true` among the keyword arguments, the sentence-transformers and HuggingFace transformers libraries follow their documented behavior: they dynamically import the `auto_map` Python module referenced in the model repository's `config.json`. That import executes arbitrary module-level Python code inside the ChromaDB server process. The kwargs validation present in the embedding-function code path explicitly permits booleans, so the dangerous flag passes through to the underlying `AutoModel.from_pretrained()` call without any sanitization [2][13].

The exploit, from the attacker's perspective, requires only a single HTTP request — after staging a malicious model repository on HuggingFace. A single unauthenticated POST request to an exposed ChromaDB endpoint, referencing an attacker-controlled HuggingFace repository that contains a benign-looking `config.json` and a malicious `modeling_*.py` payload, causes the target server to download and execute that payload before returning a 403 response. The 403 is delivered after the compromise has already occurred, which means defenders relying on access logs alone will see what appears to be a rejected anonymous request rather than a successful exploitation. HiddenLayer's analysis describes the underlying design assumption as the treatment of "embedding-function instantiation as cheap parsing rather than as a security-relevant action" [2].

The Compromise Yield

Successful exploitation grants code execution in the context of the ChromaDB server process, which in typical deployments is also the process that holds credentials for the embedding provider (OpenAI, Cohere, or a self-hosted model), the cloud storage layer where indexes are persisted, and any observability or telemetry backend wired into the deployment. The HuggingFace cache directory and process environment together yield API keys, mounted Kubernetes secrets, and any files reachable on disk [1][3][5].

The blast radius extends through the application layer. RAG pipelines that inject retrieved passages directly into LLM context windows — a common pattern — are exposed to adversarial content if an attacker modifies the underlying collections. An attacker with code execution on the vector store can poison retrieved content, modify collections to inject adversarial documents into future queries, or simply exfiltrate the embedded corpus. The compromise therefore creates the potential for adversarial influence over downstream LLM responses that depend on that retrieval substrate — particularly if the attacker inserts adversarial documents into collections before the compromise is detected — a class of impact the CSA *Securing LLM Backed Systems* guidance specifically warned about under the heading of indirect prompt injection and vector store integrity [11].

A Pattern, Not an Incident

ChromaToast is one of several severe AI infrastructure vulnerabilities disclosed in the first half of 2026, and the pattern is more instructive than any single defect. Bleeding Llama (CVE-2026-7482), disclosed by Cyera in May 2026, is an unauthenticated heap out-of-bounds read in the GGUF model loader used by Ollama; three anonymous API calls expose the entire process heap, including system prompts, user messages, environment variables, and any API tokens the process holds, and Cyera estimated more than 300,000 Ollama servers were exposed to the public internet at disclosure [6][14]. In March 2026, a cluster of CVEs in LangChain and LangGraph covered path traversal (CVE-2026-34070), deserialization of untrusted data (CVE-2025-68664) that leaks API keys and secrets, and SQL injection in the LangGraph SQLite checkpoint implementation (CVE-2025-67644) [7]. MLflow has accumulated a long sequence of critical defects across 2024 and 2025, including directory-traversal RCE in the tracking server (CVE-2025-11201), command injection in the model-serving container (CVE-2025-15379), and a cloudpickle-based deserialization RCE in `mlflow.pyfunc.load_model` (CVE-2024-37054) [8][9][15].

Three structural features explain why the pattern repeats. AI infrastructure projects optimize for developer experience and time-to-prototype, which means defaults that omit authentication, accept rich object configurations from clients, and download and execute external artifacts as ordinary operations.

The libraries this layer depends on — HuggingFace transformers, pickle, cloudpickle, GGUF — were not designed for adversarial inputs, and the HuggingFace `trust_remote_code` mechanism is a documented sharp edge that any project receiving model identifiers from an untrusted caller inherits as an arbitrary code execution sink [16][17]. The projects themselves are young and maintained by small teams under heavy adoption pressure; the ChromaToast disclosure timeline — five months of unanswered reports, no patch at public disclosure — reflects a failure of the vendor response process, whether through resource constraints, triage error, or process breakdown.

Why the AI Application Tier Is Different

Remote code execution on AI application infrastructure differs from data-layer compromise: an attacker gains not only the data stored in the service but also the credentials with which the AI system acts, and the ability to manipulate the content retrieved by language models — compounding the impact across the application layer. A compromised ChromaDB instance is not only a data breach; it is a vector for prompt injection against every RAG-backed assistant served by the application. A compromised Ollama instance exposes the heap contents of the server process at the time of exploitation — which may include system prompts, user messages, and API tokens for sessions currently or recently resident in memory. A compromised LangChain deployment yields the credentials with which the agent acts in cloud APIs and SaaS systems. CSA's *State of Non-Human Identity and AI Security* report documented that more than half of organizations operate AI agents under generic workload identity and that 43 percent share service accounts across agent functions, both of which expand the radius of any single compromise of this tier [18].

The orchestration layer compounds the problem. An AI agent that uses LangChain to query a vector database to assemble context for an LLM call sits on top of three independent infrastructure projects, each with its own credential set, each shipped with permissive defaults, each capable of exposing the others. CSA's MAESTRO threat-modeling framework explicitly treats this composition — the agentic stack of model, retrieval, orchestration, and tools — as a layered attack surface in which compromise of any single layer propagates upward [19].

Recommendations

Immediate Actions (0-7 days)

Organizations running the ChromaDB Python FastAPI server should treat any internet-exposed instance, and any instance reachable from an untrusted internal network segment, as potentially compromised until confirmed otherwise. The first containment step is to restrict network access to the ChromaDB API port to a defined allowlist enforced at the network layer rather than at the application layer; this prevents the vulnerable code path from being reachable by anonymous callers while the upstream defect remains unpatched. Where the deployment topology allows, organizations should switch to the Rust frontend (`chroma run`), which is not affected by CVE-2026-45829 [1][2][3].

Because the exploit grants full process privileges, hosts that may have been reached by anonymous external callers since November 2025 should be treated as compromised endpoints. Credentials reachable from those hosts should be rotated, including the embedding-provider API keys held in environment variables, cloud storage credentials, model-provider keys for any LLM the RAG pipeline calls into, and any Kubernetes service-account tokens mounted into the pod. Forensic indicators include unauthorized directories under `~/.cache/huggingface/modules/transformers_modules/`, unexpected outbound connections to `huggingface.co` from the ChromaDB host, and 403 responses on the `POST /api/v2/.../collections` endpoint that immediately precede unexpected outbound activity from the same process [2][12].

Operators should additionally validate the integrity of the vector store itself. An attacker with code execution can insert adversarial documents into existing collections, alter embeddings, or replace collection-level configurations to redirect future queries. Reconstruction from a known-good source corpus is the most reliable remediation; comparison of current embedding distributions against a recent backup is a useful detection step where reconstruction is not immediately feasible.

For organizations operating Ollama, LangChain, LangGraph, or MLflow in production, the same posture is warranted. Bleeding Llama is patched in Ollama 0.17.1 but the patch was not flagged as a security release; operators should verify their installed version and apply the upgrade regardless of whether their release notes flagged the change as security-relevant [6][14]. The LangChain and LangGraph CVEs disclosed in March 2026 have patched versions available and should be applied through normal vulnerability management cadence; MLflow operators should ensure they are on the post-3.8.2 line that includes the shlex hardening for CVE-2025-15379 [7][8].

Short-Term Mitigations (7-90 days)

AI infrastructure components should be brought into the same vulnerability management, asset inventory, and configuration baseline programs that govern traditional databases and application servers. Several specific practices follow from the ChromaToast incident and the surrounding pattern.

The first is authenticating the AI infrastructure tier rather than treating its defaults as acceptable. Both the ChromaDB Python server and Ollama are documented as shipping without required authentication; a reverse proxy that enforces mTLS or OIDC in front of the API, rather than relying on the application's own auth model, gives defenders an enforcement point that is not subject to ordering bugs in the application code path. This proxy pattern would have blocked the specific unauthenticated vector in CVE-2026-45829, provided no API paths bypass the authentication layer.

The second is restricting outbound network access from AI infrastructure hosts to a narrow allowlist. Both the ChromaToast and the historical HuggingFace pickle-execution incidents depend on the target process being able to reach `huggingface.co` or another external model registry to retrieve attacker-supplied code [16][17]. Egress filtering to a curated mirror, or an internal HuggingFace mirror behind an artifact promotion process, removes a class of exploitation primitives even when the upstream defect is not yet patched.

The third is treating model artifacts as code. Models referenced by name from external registries should not be loaded into production processes without artifact-scanning and provenance verification. CSA's *Securing LLM Backed Systems* guidance treats vector stores and the model-serving substrate as security-relevant in their own right, not merely as supporting infrastructure to the LLM; the practical implication is that the model-loading path requires the same controls applied to package installation, including signed artifacts, scanned content, and a controlled promotion path from external registries [11].

The fourth is scoping the credentials available to AI infrastructure processes to the minimum required for the workload, with preference for short-lived federated credentials over long-lived service-account keys. CSA's *State of Non-Human Identity and AI Security* findings make clear that this is not current practice in most organizations; the ChromaToast yield — environment variables, mounted secrets, all files on disk — is large in part because many ChromaDB hosts in production are believed to retain credential sets sized for development convenience, given the project's default-open ergonomics — a provisioning pattern that expands the yield of any successful compromise.

Strategic Considerations

The AI Controls Matrix (AICM) provides the canonical control set for AI system assurance, with particular relevance here in the Supply Chain Management domain, the Identity and Access Management domain that governs authentication boundaries on inference and retrieval services, and the Infrastructure and Networking Security domain that addresses the exposure profile of AI application components [20][21]. Organizations whose AI infrastructure inventory exists in pockets across data science, platform engineering, and individual application teams should consolidate it under a single AICM-aligned assessment. ChromaToast is a case where the controls articulated in the AICM — explicit authentication on AI service endpoints, restricted egress on model-loading paths, vetted external dependencies — would have materially reduced or eliminated the exploitation path.

The MAESTRO threat-modeling framework treats the agentic stack as distinct layers with their own trust boundaries; in the ChromaToast case the failed boundary was between the orchestration request and the retrieval substrate [19]. Threat models for AI applications should explicitly enumerate AI infrastructure components in scope and require that each layer enforce its own authentication rather than inheriting trust from the layer above. For consumers of AI services delivered by third parties, STAR for AI provides an assurance mechanism through which an organization can confirm that providers have implemented the relevant controls [22]; the growing population of services built on ChromaDB, Ollama, LangChain, and MLflow makes the question of provider-side patching a routine due diligence item.

Finally, organizations should plan for this layer to behave more like the early years of an open-source ecosystem — high vulnerability discovery rates, slow vendor response, frequent zero-days — than like the mature database or web-server stacks they are accustomed to. ChromaToast is the third significant unauthenticated vulnerability disclosed against an AI infrastructure component in 2026 alone. Patch availability has varied: Ollama's Bleeding Llama received a fix in 0.17.1, while ChromaDB had no patch at the time of public disclosure despite a five-month disclosure window. Organizations should not assume timely upstream remediation and should invest in compensating controls — network-layer controls, credential isolation, and detective monitoring — that remain effective in the absence of patches.

CSA Resource Alignment

This research note aligns with several active CSA frameworks and prior publications. The AI Controls Matrix is the canonical control set for the issues raised here, with the Supply Chain Management, Identity and Access Management, Infrastructure and Networking Security, and Threat and Vulnerability Management domains all bearing directly on the ChromaToast exposure profile [20][21]. The MAESTRO

threat-modeling framework provides the layered model in which the retrieval substrate is a distinct attack surface within the agentic stack and supplies the language for threat modeling that surfaces issues like CVE-2026-45829 before they ship [19].

CSA's *Securing LLM Backed Systems: Essential Authorization Practices* report, published by the AI Technology and Risk Working Group in 2024, treated vector database security and RAG architecture as discrete security domains and articulated authorization patterns that the ChromaToast incident vindicates [11]. The *State of Non-Human Identity and AI Security* report supplies the empirical baseline for the credential-scope recommendations above, including the prevalence of generic workload identity and shared service accounts in AI deployments [18]. The Cloud Controls Matrix v4.1 remains the authoritative control set for the underlying cloud and platform layer that hosts these AI infrastructure components, and STAR for AI provides the consumer-side verification mechanism for organizations that depend on AI services rather than operating their own infrastructure [22][23].

For organizations that operate AI agents on top of compromised or vulnerable retrieval substrates, the *Agentic AI Red Teaming Guide* describes test patterns that surface the cascading effects of an infrastructure compromise on downstream agent behavior; this is a useful scoping reference for post-incident assurance work [24].

References

[1] HiddenLayer. "ChromaToast Served Pre-Auth." HiddenLayer Research, May 13, 2026.

[2] Hadrian. "CVE-2026-45829 – ChromaDB Python server hands you RCE before it asks who you are." Hadrian Blog, May 19, 2026.

[3] BleepingComputer. "Max-severity flaw in ChromaDB for AI apps allows server hijacking." BleepingComputer, May 2026.

[4] PyPI. "chromadb · PyPI." Python Package Index, 2026.

[5] SecurityWeek. "Unpatched ChromaDB Vulnerability Can Lead to Server Takeover." SecurityWeek, May 2026.

[6] Cyera. "Bleeding Llama: Critical Unauthenticated Memory Leak in Ollama." Cyera Research, May 2026.

[7] The Hacker News. "LangChain, LangGraph Flaws Expose Files, Secrets, Databases in Widely Used AI Frameworks." The Hacker News, March 2026.

[8] ZeroPath. "MLflow Tracking Server CVE-2025-11201: Brief Summary of Directory Traversal Remote Code Execution." ZeroPath Blog, 2025.

[9] SentinelOne. "CVE-2025-15379: MLflow Command Injection RCE Vulnerability." SentinelOne Vulnerability Database, 2025.

[10] Chroma. "Chroma raises $18M seed round." Chroma, 2024.

[11] Cloud Security Alliance. "Securing LLM Backed Systems: Essential Authorization Practices." CSA AI Technology and Risk Working Group, 2024.

[12] National Vulnerability Database. "CVE-2026-45829." NIST NVD, May 2026.

[13] HuggingFace. "Security considerations for trust_remote_code=True." HuggingFace Diffusers Discussion #12033, 2025.

[14] The Hacker News. "Ollama Out-of-Bounds Read Vulnerability Allows Remote Process Memory Leak." The Hacker News, May 2026.

[15] National Vulnerability Database. "CVE-2024-37054." NIST NVD, 2024.

[16] CSO Online. "Attackers hide malicious code in Hugging Face AI model Pickle files." CSO Online, 2025.

[17] Unit 42 (Palo Alto Networks). "Remote Code Execution With Modern AI/ML Formats and Libraries." Unit 42, 2025.

[18] Cloud Security Alliance. "The State of Non-Human Identity and AI Security." CSA, 2026.

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

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

[21] Cloud Security Alliance. "Introductory Guidance to AICM." CSA, 2025.

[22] Cloud Security Alliance. "STAR for AI." CSA, 2026.

[23] Cloud Security Alliance. "CCM v4.1: Strengthening Cloud Security." CSA, December 2, 2025.

[24] Cloud Security Alliance. "Agentic AI Red Teaming Guide." CSA, May 2025.