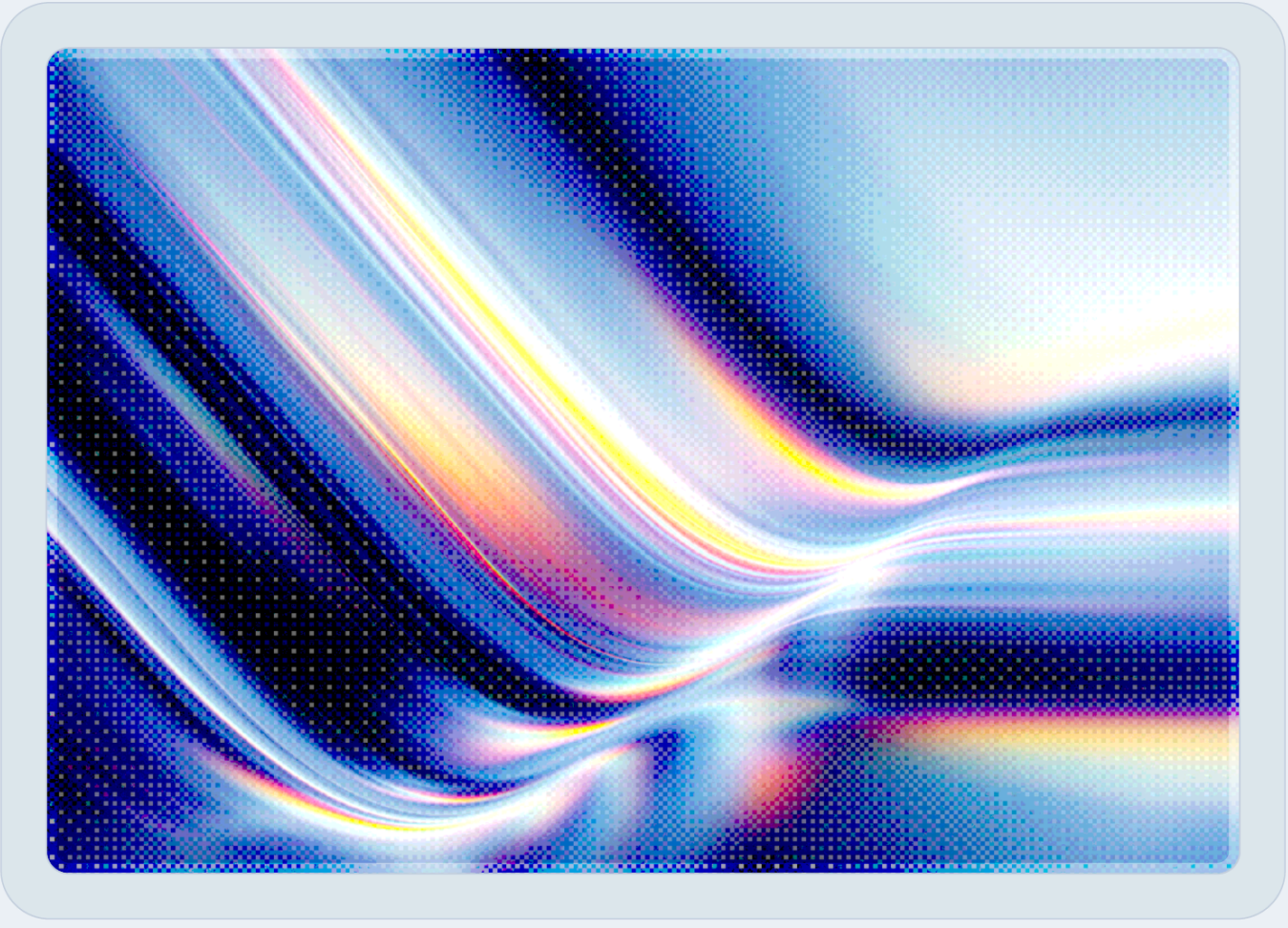
CSAI Foundation | Cloud Security Alliance

ChromaDB RCE Exposes Unauthenticated AI Infrastructure

CVE-2026-45829 (ChromaToast): Analysis and Guidance for AI Security Teams

2026-05-21

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

- CVE-2026-45829 ("ChromaToast") is a CVSS 10.0 (maximum severity) pre-authentication remote code execution vulnerability in ChromaDB's Python FastAPI server, affecting all versions from 1.0.0 through at least 1.5.8, with no official patch issued as of May 2026 [1][2].
- An unauthenticated attacker with HTTP access to ChromaDB's API port can achieve full server compromise by supplying a malicious HuggingFace model reference in a collection creation request, gaining a shell with the privileges of the database process and access to all data, secrets, and credentials the server can reach [3].
- ChromaDB is downloaded approximately 13–14 million times per month and is the default vector store in widely-used AI application frameworks including LangChain and LlamaIndex; an estimated 73% of internet-exposed instances are running a vulnerable version [1][3][4].
- The vulnerability was first reported to Chroma maintainers prior to HiddenLayer's February 2026 disclosure, according to CVE advisory records [9]; HiddenLayer formally reported it on February 17, 2026, with multiple follow-up attempts through April 2026 receiving no response, leading to coordinated public disclosure on May 18, 2026 [3].
- Organizations should immediately verify that ChromaDB Python FastAPI servers are not exposed to untrusted networks. Where exposure exists, deploying a network-level authentication proxy or migrating to the Rust-based `chroma run` frontend are the primary mitigations available until an official patch is released.

Background

ChromaDB is an open-source vector database purpose-built for AI applications. It provides persistent storage and fast approximate nearest-neighbor search over high-dimensional embeddings, making it a common data layer for retrieval-augmented generation (RAG) architectures, semantic search systems, and AI memory stores. The project's Python client library integrates natively with LangChain, LlamaIndex, Haystack, and the OpenAI embeddings API, reflecting its adoption across a wide spectrum of both prototyping and production AI deployments [4][5].

ChromaDB exposes its functionality through two distinct server frontends. The original Python server, built on FastAPI, handles HTTP requests and serves the REST API used by client applications. A newer Rust-based frontend, invoked via the `chroma run` command, was introduced as a more performant alternative. The CVE-2026-45829 vulnerability exists exclusively in the Python FastAPI server. Deployments using `chroma run` and official ChromaDB Docker images are not affected by this specific flaw [3][6].

The vulnerability was independently identified prior to HiddenLayer's public disclosure, according to CVE and advisory records [9]. HiddenLayer, a company specializing in AI security research, formally reported the issue on February 17, 2026, following the responsible disclosure process documented on Chroma's security page. Subsequent follow-up attempts on February 24, March 5, and April 16 — through multiple email channels, IT-ISAC, and social media — received no response from Chroma maintainers [3][7]. With CVE-2026-45829 formally reserved on May 12, 2026, HiddenLayer published full public disclosure on May 18, 2026 [3][7]. As of ChromaDB 1.5.8, the vulnerability remains unpatched [2][7].

Security Analysis

Root Cause: Authentication After Execution

The vulnerability arises from a fundamental sequencing error in ChromaDB's FastAPI request handling. The critical code path lives in `chromadb/server/fastapi/__init__.py`, specifically in the `create_collection` handler. When a client sends a collection creation request, the server parses the raw JSON body and immediately invokes `load_create_collection_configuration_from_json()` on the attacker-controlled configuration blob — before any authentication check is performed [6]. This function instantiates the embedding function named in the configuration, and for the `sentence_transformer` embedding function, the `build_from_config` method forwards the caller's keyword arguments directly into the underlying model loader without sanitization.

The practical consequence is that two attacker-controlled parameters — `model_name` (pointing to an arbitrary HuggingFace repository) and `trust_remote_code: true` (enabling execution of Python code from that repository) — are sufficient to trigger arbitrary code execution on the server host before the server has confirmed who sent the request. No credentials, API keys, or prior knowledge of the server's configuration are required [3][6].

Attack Surface and Impact

The exploitation technique is straightforward and requires only an HTTP request to the ChromaDB API port. A proof-of-concept exploit, confirmed by HiddenLayer, demonstrates that an attacker can:

Impact Category	Details
Credential theft	API keys, OAuth tokens, database connection strings in environment variables
Secret exfiltration	Kubernetes secrets, mounted credential files, service account tokens
Data access	All vector collections, stored documents, and embeddings
Lateral movement	Full shell access enabling network reconnaissance and pivoting
Persistence	Potential ability to install backdoors or modify server behavior; dependent on attacker post-exploitation actions

An attacker achieving RCE through this path inherits the full privileges of the ChromaDB process. In containerized deployments this is typically a service account; in under-hardened deployments, it may be root or a cloud identity with broad IAM permissions. Because ChromaDB is frequently deployed as part of an AI application stack alongside LLM inference endpoints, retrieval pipelines, and API gateways, a compromised ChromaDB instance gives an attacker access to the data flowing through an entire AI application.

Disclosure Timeline

Date	Event
November 2025	Vulnerability independently identified and reported to Chroma prior to HiddenLayer's disclosure; no response received [9]
February 17, 2026	HiddenLayer formally reports CVE-2026-45829 to Chroma via security disclosure page

Date	Event
February 24, 2026	HiddenLayer follow-up via alternate email channels; no response
March 5, 2026	HiddenLayer attempts contact through IT-ISAC; no response
April 16, 2026	Final follow-up through all prior channels and social media; no response
May 12, 2026	CVE-2026-45829 reserved
May 18, 2026	HiddenLayer publishes full public disclosure
May 2026	ChromaDB 1.5.9 released; no official patch confirmed in primary sources [2][7]

Why This Vulnerability Is Particularly Consequential

The combination of factors driving CVE-2026-45829's severity is unusual even for a CVSS 10.0 flaw. ChromaDB's installation footprint — approximately 13–14 million monthly pip downloads — places vulnerable code in a substantial fraction of the AI developer ecosystem [1][3]. Its role as the default vector store in LangChain and LlamaIndex tutorials means that many developers stand up ChromaDB instances following introductory documentation that may not emphasize network isolation as a prerequisite for production deployment. The Python FastAPI server has historically been the most accessible path for getting ChromaDB running quickly, which contributes to its prevalence in environments where hardening may have been deferred.

The HuggingFace model loading mechanism that enables this exploit reflects a broader pattern in AI infrastructure: the trust model inherited from research and experimentation environments — where arbitrary model downloading is normal and expected — conflicts with production security requirements. When `trust_remote_code: true` is an easily supplied parameter and HuggingFace model loading is a first-class operation, the attack surface extends well beyond ChromaDB itself to any framework that exposes similar mechanisms without authentication gates.

The absence of a vendor response over a six-month disclosure window, culminating in forced public disclosure, raises additional concerns about the security posture of the ChromaDB project. Organizations that have built production systems on ChromaDB must plan for a prolonged period without an official patch and implement compensating controls accordingly.

Recommendations

Immediate Actions

Security teams should treat CVE-2026-45829 as requiring immediate response for any internet-accessible ChromaDB deployment. The first priority is exposure verification: organizations should audit their environment to determine whether any ChromaDB Python FastAPI server is reachable from untrusted networks, including internal corporate networks without appropriate segmentation. Shodan and similar tools have indexed internet-exposed ChromaDB instances [1]; assuming no exposure exists without verification leaves an unquantified risk.

Where exposed instances are found, the immediate control is network-level isolation. The ChromaDB API port should be placed behind a firewall rule permitting access only from application servers that legitimately need it. In Kubernetes environments, NetworkPolicy resources should restrict ingress to the ChromaDB service to the specific pod selectors of authorized clients. These controls should be treated as emergency measures, not permanent solutions, because they do not address the underlying vulnerability.

Short-Term Mitigations

For teams that cannot immediately migrate away from the Python FastAPI server, deploying an authenticating reverse proxy in front of ChromaDB provides a meaningful compensating control. An nginx, Caddy, or cloud-native API gateway configured with credential verification at the network boundary prevents unauthenticated requests from ever reaching the vulnerable code path. This approach does not remediate the vulnerability but raises the bar for exploitation substantially above the current unauthenticated HTTP request level.

The preferred near-term mitigation for production deployments is migration to the Rust-based `chroma run` frontend, which is confirmed not affected by CVE-2026-45829 [3]. Teams should evaluate whether their application's API usage patterns are compatible with the Rust frontend and prioritize that migration if so. Similarly, official ChromaDB Docker images ship with a configuration that avoids the vulnerable code path; verifying that production deployments use these images rather than custom Python server invocations is a lower-effort action for containerized environments.

Where HuggingFace model loading is not a required capability, organizations should audit their ChromaDB configuration and application code to determine whether `trust_remote_code` can be disabled or whether the `sentence_transformer` embedding function can be replaced with one

that does not invoke the vulnerable code path. Restricting outbound network access from ChromaDB containers to known-good HuggingFace endpoints, or blocking it entirely, adds a layer of defense-in-depth that limits what a successfully delivered exploit payload can do.

Strategic Considerations

The vulnerability's impact suggests that a meaningful share of ChromaDB deployments may have been configured as research conveniences rather than production infrastructure components, potentially resulting in a lower security baseline than organizations apply to relational data stores [10]. Security architects should conduct an explicit review of all vector database deployments against the same control baseline applied to PostgreSQL or other data-tier services. CVE-2026-45829 illustrates that vector databases occupy the same threat model as other sensitive data stores — they require authentication, network isolation, secrets management, and a monitored patching lifecycle.

The six-month period between initial disclosure and forced public disclosure — with no response to multiple contact attempts — is also a signal that organizations must independently track vulnerability status for open-source AI infrastructure components that may not have formal security response processes. Subscribing to GitHub Security Advisories for core AI dependencies, integrating PyPI/npm vulnerability scanning into CI pipelines, and establishing explicit SBOM coverage for AI-specific libraries are foundational practices that would have surfaced this issue as an unresolved risk before public disclosure.

Finally, the mechanism of this exploit — a model artifact sourced at runtime from an external registry triggering code execution — is a pattern that is likely to recur across AI infrastructure as model loading becomes a standard application operation. Security teams should begin treating HuggingFace model identifiers with the same scrutiny applied to container image tags: provenance verification, hash pinning, and controlled access to model registries are supply-chain controls that belong in AI application security requirements.

CSA Resource Alignment

CVE-2026-45829 maps directly to threat categories identified in CSA's MAESTRO framework for agentic AI threat modeling [11]. At Layer 5 (Infrastructure and Orchestration), MAESTRO identifies unauthorized access to data stores and compute infrastructure as a primary risk vector for AI systems — the exact class of attack that ChromaToast enables. The lack of authentication on the ChromaDB

Python server represents a direct failure at this layer, and MAESTRO's corresponding controls — network segmentation, access control enforcement, and monitoring of infrastructure APIs — are the right lens for evaluating and remediating this exposure.

At Layer 3 (Agent Memory and Context), ChromaDB stores the retrieved context that shapes LLM responses in RAG applications. A compromised ChromaDB instance gives an adversary the ability to read, modify, or poison the vector store, enabling downstream context manipulation attacks against the AI systems that rely on it. This connects to MAESTRO's data integrity and context poisoning threat categories, and reinforces why vector database security is inseparable from the security of the AI applications they support.

The CSA AI Controls Matrix (AICM) provides a practical control framework for remediating the gaps this vulnerability exposes. AICM controls addressing infrastructure access management (IAM), data-at-rest protection, and secure configuration of AI supporting services should be reviewed and updated to include explicit requirements for vector database authentication and network isolation. Organizations using the STAR Registry to document their AI security posture should reflect the current unpatched status of ChromaDB deployments as an open risk item until a confirmed fix is available.

CSA's Zero Trust guidance is directly applicable: the absence of any authentication on ChromaDB's Python server directly contradicts the zero trust principle that no request should be trusted based solely on network location. Enforcing identity verification at every service boundary — including internal data-tier APIs — prevents the implicit trust assumption that this vulnerability exploits. Cloud security teams implementing zero trust architectures should extend that model explicitly to AI infrastructure components, which have often been treated in practice as internal services exempt from the same verification requirements applied to user-facing APIs. This exposure demonstrates why that assumption cannot be sustained in production AI application stacks.

References

[1] Toulas, B. "Max-severity flaw in ChromaDB for AI apps allows server hijacking." BleepingComputer, May 19, 2026.

[2] Arghire, I. "Unpatched ChromaDB Vulnerability Can Lead to Server Takeover." SecurityWeek, May 19, 2026.

[3] Tonglet, E. "ChromaToast Served Pre-Auth." HiddenLayer, May 18, 2026.

[4] SC Media. "Max-severity vulnerability in ChromaDB allows unauthenticated remote code execution." SC Media, May 2026.

[5] CXO Digital Pulse. "Unpatched ChromaDB Vulnerability Could Allow Full Server Takeover." CXO Digital Pulse, May 2026.

[6] Hadrian Security Research. "CVE-2026-45829 — ChromaDB Python server hands you RCE before it asks who you are." Hadrian, May 2026.

[7] CyberExpress. "CVE-2026-45829: ChromaDB FastAPI ChromaToast RCE Exploit Now." The Cyber Express, May 2026.

[8] NIST National Vulnerability Database. "CVE-2026-45829 Detail." NVD, 2026.

[9] GitHub Security Advisory Database. "GHSA-f4j7-r4q5-qw2c." GitHub, 2026.

[10] Cisco Systems. "Securing Vector Databases." Cisco Security, 2024.

[11] Cloud Security Alliance. "MAESTRO: Multi-Agent Environment Security, Trust, and Risk Analysis Over sight." CSA AI Safety Initiative, 2024.

