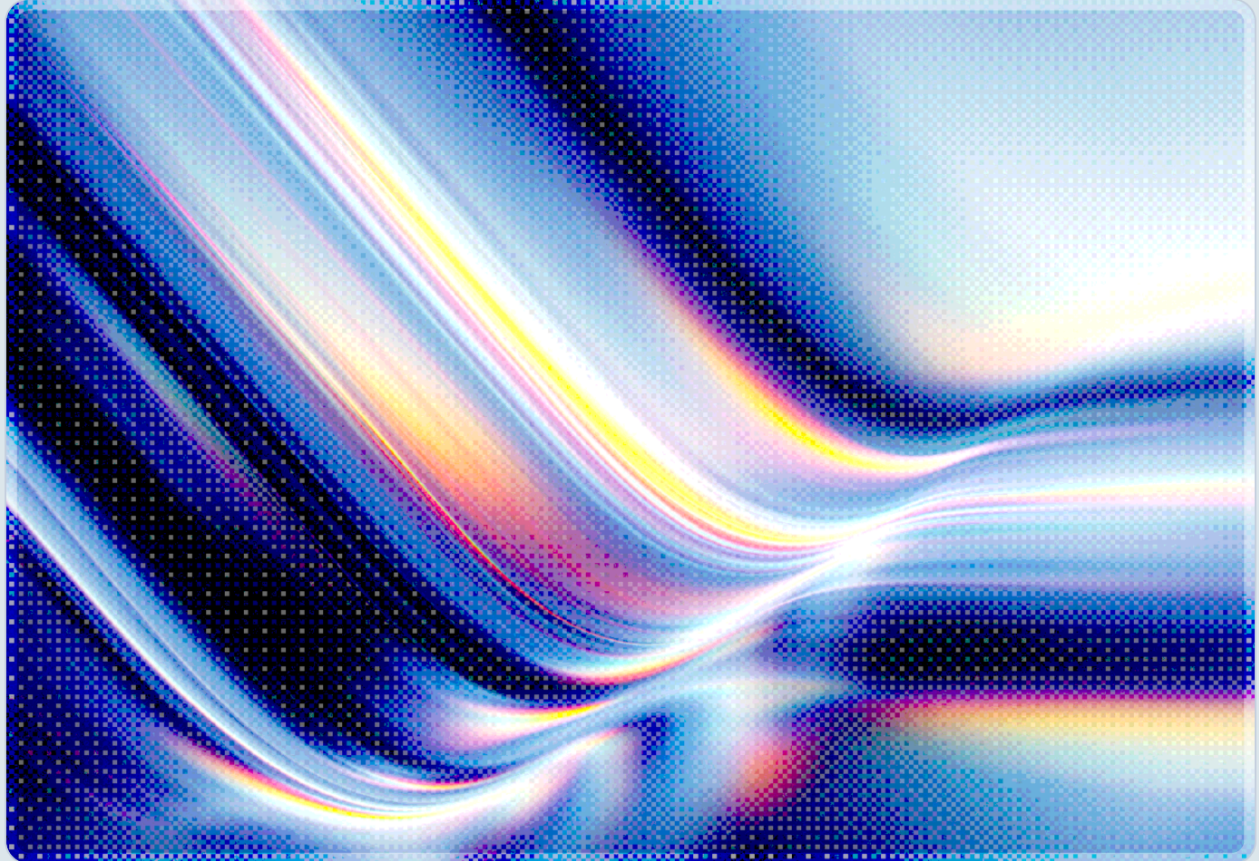


AI Training Datasets as a Secrets-Exposure Vector

2026-10-01



AI-assisted Rapid Research



CSAI

cloud
CSA security
alliance®

© 2026 Cloud Security Alliance. Some rights reserved.

You may download, store, display, view, print, redistribute, and link to this document in its original, unmodified form, provided that attribution to the Cloud Security Alliance is maintained and all trademark and copyright notices remain intact.

This document may not be modified or altered. You may quote portions of the document as permitted by the Fair Use provisions of the United States Copyright Act, provided that attribution is given to the Cloud Security Alliance.

This document may be shared on professional and social media platforms in its original form with attribution.

This document was generated with AI assistance and has not undergone official CSA review and approval processes.

Key Takeaways

A series of large-scale scans conducted throughout 2025 and 2026 has established that the public code and web corpora used to train large language models carry hundreds of thousands of live, authenticating credentials, and that almost none of them get revoked once discovered. Truffle Security's July 2026 scan of The Stack v3, a 224.5-million-repository, 5-trillion-token code dataset maintained by the BigCode project on Hugging Face, found 543,699 unique credentials that still authenticated against their origin services, with a median exposure window of 784 days and some credentials dating to 2009 [1]. A separate scan completed in June 2026 across 7.6 petabytes of Hugging Face's own hosted training datasets found 221,303 additional live credentials, including hundreds of GitHub tokens with write access and nearly 400 GB of exposed personal data [2]. An earlier scan of the Common Crawl archive used to train DeepSeek and numerous other foundation models found 11,908 live secrets across 2.76 million web pages as far back as February 2025, establishing that this is a multi-year, cross-dataset pattern rather than a single incident [3]. Industry-wide telemetry corroborates the trend at scale: GitGuardian's 2026 secrets-sprawl report counted 29 million newly hardcoded secrets on public GitHub in 2025 alone, with leaks tied specifically to AI services up 81% year over year, and found that 64% of secrets confirmed valid in 2022 remain exploitable four years later [4]. For organizations building, fine-tuning, or simply relying on models trained on public corpora, this is no longer a theoretical supply-chain concern; it is a standing, quantifiable exposure that existing push-protection and scanning tools have not closed.

Background

The datasets that train today's code-generation and general-purpose language models are built by crawling and aggregating enormous volumes of public code and web content. The scan results examined in this note suggest that secret scanning has not historically been prioritized to the same degree as deduplication, licensing filtration, and content-quality filtering in the pipelines maintained by the BigCode project, Hugging Face, and the Common Crawl Foundation, though none of the three organizations has publicly detailed its relative investment in each area. The Stack v3, the dataset at the center of Truffle Security's July 2026 findings, closed its crawl on August 7, 2025, and spans 58.4 billion files pulled from 224.5 million public GitHub repositories [1]. Truffle Security's researchers did not merely pattern-match for credential-shaped strings; they ran each discovered secret through the originating

provider's own authentication endpoint using their open-source TruffleHog scanner, so every figure cited in their research reflects a credential that was live and functional at the time of testing in July 2026, not merely one that looked like a credential [1].

The timeline behind those 543,699 live credentials is itself instructive. GitHub began offering free secret-scanning alerts to public repository owners on February 28, 2023, made push protection generally available in May 2023, and enabled push protection by default for all public repositories on February 29, 2024 [1]. Despite this, Truffle Security found that 45.2% of the live credentials it identified were committed before the free alerting program even existed, 18% were committed during the window when protection was available but optional, and 36.8% were committed after default push protection had already been enabled, meaning more than a third of today's live, exposed secrets were leaked after GitHub had already deployed automated detection at the point of commit [1]. Compounding the problem, 51.8% of the live credentials Truffle Security found used formats that GitHub's default push-protection patterns do not recognize, meaning detection at commit time would not have caught them even had the developer been using every available safeguard [1]. The consistent finding across all four research efforts referenced in this note, spanning GitHub, Hugging Face's own dataset hosting, and Common Crawl, is that detection tooling has improved considerably while revocation has not: as Truffle Security put it, "what keeps a leaked credential alive is revocation, not the block at push time" [1].

This same pattern recurs independent of which corpus is scanned. The June 2026 scan of Hugging Face's dataset hosting infrastructure, 19 times larger in volume than Truffle Security's previous largest scan, found 221,303 live credentials spread across 6,003 individual datasets, including 742 live OpenAI API keys, 26 Anthropic API keys, 349 GitHub personal access tokens, 223 of them carrying full repository write access capable of pushing code into downstream software and the remainder split across narrower scopes such as CI-workflow rewrite, organization administration, or package publishing, 318 Docker Hub tokens capable of pushing container images, and 8,557 Google Cloud service-account keys [2]. The same research estimated that credentials discovered in that scan alone exposed at least 393 GB of personal data, and calculated that the exposed AI-provider keys alone represented at minimum \$920,000 per year in potential inference-cost theft based on default spending limits [2]. The earlier Common Crawl scan, covering the December 2024 snapshot used to train DeepSeek and comparable models, found that 63% of the 11,908 live secrets it identified were reused across multiple pages, with one API key for the WalkScore service appearing 57,029 times across 1,871 subdomains, illustrating how a single leaked credential can be ingested into a training corpus many times over [3].

Security Analysis

The direct risk is that credentials embedded in training data remain live, authenticating secrets regardless of whether a model is ever trained on them, and any party capable of downloading or re-scanning the same public dataset, which by definition includes any adversary as easily as any researcher, can extract and use them immediately. Truffle Security's breakdown of credential survival rates by provider in The Stack v3 shows how unevenly this risk is distributed, as summarized in the table below.

Credential Type	Live / Candidates Found	Live Rate
npm tokens	1 / 101,886	~0%
GitHub tokens	260 / 73,048	0.36%
Google Cloud service-account keys	69,041 / 126,963	54%
Postgres connection strings	11,465 / 12,985	88%

The near-total die-off of npm tokens reflects that provider's short default token lifetimes and aggressive revocation practices, and GitHub's own tokens fared only slightly better. At the other end of the spectrum, Google Cloud service-account keys and Postgres connection strings persist at dramatically higher rates, indicating that credential types without short default lifetimes or without an equivalent to GitHub's push-protection ecosystem survive exposure far longer once leaked [1].

A second, less direct but more durable risk concerns the models themselves. A body of academic research on language model memorization, documented as of 2023, established that larger models retain and can be made to emit verbatim fragments of their training data at higher rates than smaller ones, and that techniques for extracting such memorized sequences from production models, not just research prototypes, were improving at the time, though more recent confirmation that the trend has continued through 2026 would strengthen the point [5][6]. OWASP's Top 10 for LLM Applications formalizes this as Sensitive Information Disclosure (LLM02), noting that models can memorize and reproduce fragments of training data including credentials, personally identifiable information, and proprietary business logic, independent of any flaw in how the deployed application itself is configured [7]. The practical implication is that even a hypothetical, perfectly scrubbed downstream application built on top of a model trained on an unscrubbed corpus inherits some residual probability of regurgitating a credential it was never supposed to see again, a risk surface that application-layer controls alone are unlikely to fully mitigate, since the exposure originates upstream, in the training pipeline.

A third risk operates at one remove from direct credential theft: models trained on code containing exposed secrets learn the surrounding patterns as well as the secrets themselves. Industry researchers examining this exposure have specifically flagged the concern that code-generation assistants trained on corpora that include hardcoded credentials may normalize or reproduce the same insecure pattern, suggesting hardcoded keys or disabled validation as plausible-looking completions to developers who have no reason to suspect the suggestion reflects a known-bad practice rather than an idiomatic one [3]. This compounds the first-order exposure: it is not only that old secrets survive inside the corpus, but that the coding habits responsible for creating them in the first place may be reinforced by the very tooling organizations now use to write new code.

Finally, this is not a static or one-time exposure. In the absence of an explicit remediation or rescanning step, retraining cycles, derivative fine-tunes, and dataset re-releases will typically carry forward whatever secrets were present at the time of the original crawl, and the scale of these corpora, petabytes of data spanning hundreds of millions of files, means that manual review alone is unlikely to be a sufficient mitigation path at this scale. In CSA's assessment, the dataset curators themselves (BigCode, Hugging Face, and the Common Crawl Foundation) are best positioned to close this gap, followed by the enterprises that fine-tune on derivative or internally-collected corpora that inherit the same blind spot.

Recommendations

Immediate Actions

Security teams should treat any credential discoverable by searching public code or web archives as already compromised, regardless of whether there is direct evidence of misuse, and prioritize revocation and rotation over detection alone. Organizations should run their own secret-scanning tooling, such as TruffleHog or an equivalent verified-secret scanner, against any internal datasets, forks, or derivative corpora built from public sources such as GitHub mirrors, Hugging Face datasets, or Common Crawl snapshots before using them for fine-tuning or retrieval-augmented pipelines. Teams that have previously committed credentials to public repositories, even years ago and even if GitHub's alerts were never enabled at the time, should assume those credentials may now be embedded in one or more downstream training corpora and rotate them rather than relying on the original repository's deletion or privatization, since dataset snapshots taken before a remediation persist independently of the source repository's current state.

Short-Term Mitigations

Organizations fine-tuning models on internally curated or scraped data should insert a verified-secret-scanning stage into the data pipeline itself, positioned before training rather than as a post hoc audit, and should configure that stage to flag and quarantine files rather than merely log findings, given how consistently detection without a forced remediation step has failed to translate into revocation across the datasets examined here. Given that over half of the live credentials found in The Stack v3 used formats unrecognized by GitHub's default push-protection patterns, teams should not treat push-protection coverage as comprehensive and should supplement it with broader-pattern or entropy-based scanning tuned to the specific credential types relevant to their own cloud and SaaS footprint. Procurement and AI-governance teams evaluating third-party models or fine-tuning services should ask vendors directly what secret-scanning and remediation steps, if any, are applied to training corpora before use, since this is not yet a standardized disclosure and currently varies widely across providers.

Strategic Considerations

The credential-exposure pattern documented across The Stack v3, Hugging Face's dataset hosting, and Common Crawl is best understood as a non-human identity governance failure that happens to surface through the AI training pipeline rather than a defect unique to AI itself: the same hardcoded, unrotated, unmonitored secrets that leak into public code have always posed a risk, and large-scale dataset crawling simply aggregates and amplifies an existing credential-hygiene gap into a single, highly concentrated target. Organizations should factor this into how they weigh the provenance of any model, dataset, or fine-tune they adopt, treating "was this corpus scanned and remediated for secrets before training" as a due-diligence question on par with license compliance or data-quality review, rather than an afterthought. As dataset curators and model providers come under increasing pressure to document their training data provenance, CSA expects secret-scanning and revocation evidence to become a standard element of model and dataset transparency reporting, and organizations with mature non-human identity programs are positioned to adapt to that expectation well ahead of the rest of the industry.

CSA Resource Alignment

CSA's [State of Non-Human Identity Security Survey Report](#), based on a 2024 survey of 818 IT and security professionals, provides independent corroboration of the root cause behind this exposure pattern: respondents reported only 15% high confidence in their ability to prevent non-human-identity attacks (compared to 25% for human-identity attacks), found that 68% of GitHub tokens in their own

environments carried no expiry, and identified credential-rotation failures as a factor in 45% of confirmed non-human-identity incidents [8]. That survey's finding of an average of 4.5 distinct locations per leaked secret maps directly onto what this note documents at dataset scale: a single hardcoded credential does not leak once, it propagates across forks, mirrors, snapshots, and now training corpora, each a separate location an organization must track down to fully remediate. CSA's research into credential exposure in government cloud environments reaches a parallel conclusion from a different angle: a single leaked GitHub credential tied to a federal agency's cloud environment was found to persist, unrotated and unrevoked, well after discovery, reinforcing that static, unmonitored credentials represent a governance failure that recurs across government, enterprise, and now AI-training-data contexts alike [10]. Read together, this body of research indicates that the training-data exposure problem is a symptom of a broader non-human identity governance gap that predates, and will outlast, any single dataset-remediation effort.

The AI Controls Matrix (AICM v1.1) offers a directly applicable control structure for organizations addressing this gap, specifically through its Data Security and Privacy Lifecycle Management (DSP) domain, which governs the handling, provenance, and lifecycle protection of data used throughout the AI stack, and its Model Security (MDS) domain, which CSA designed explicitly to address the availability and integrity of training data, model weights, and the infrastructure used to develop them [9]. Organizations building internal fine-tuning pipelines or evaluating third-party datasets should map their data-ingestion and curation processes against both domains to identify where a verified-secret-scanning checkpoint belongs in their own pipeline, and where contractual or disclosure requirements should be placed on any third party supplying training data or pretrained models.

References

- [1] Truffle Security. "[GitHub Repos Exposed 543,699 Credentials. Nobody Revoked Them.](#)" Truffle Security, July 2026.
- [2] Truffle Security. "[Scanning 7.6 Petabytes of Hugging Face Training Data for Secrets.](#)" Truffle Security, June 1, 2026.
- [3] Truffle Security. "[Research finds 12,000 'Live' API Keys and Passwords in DeepSeek's Training Data.](#)" Truffle Security, February 27, 2025.
- [4] GitGuardian. "[The State of Secrets Sprawl 2026.](#)" GitGuardian, 2026.
- [5] Nasr, M., et al. "[Scalable Extraction of Training Data from \(Production\) Language Models.](#)" arXiv, November 2023.
- [6] "[SoK: Memorization in General-Purpose Large Language Models.](#)" arXiv, October 2023.
- [7] OWASP. "[OWASP Top 10 for LLM Applications 2025.](#)" OWASP Foundation, 2025.
- [8] Cloud Security Alliance. "[The State of Non-Human Identity Security.](#)" Cloud Security Alliance, June 2024.
- [9] Cloud Security Alliance. "[AI Controls Matrix \(AICM\) v1.1.](#)" Cloud Security Alliance, 2026.
- [10] Cloud Security Alliance. "[CISA GovCloud Credential Exposure: Institutional Risk to AI Cybersecurity Governance.](#)" Cloud Security Alliance, 2026.