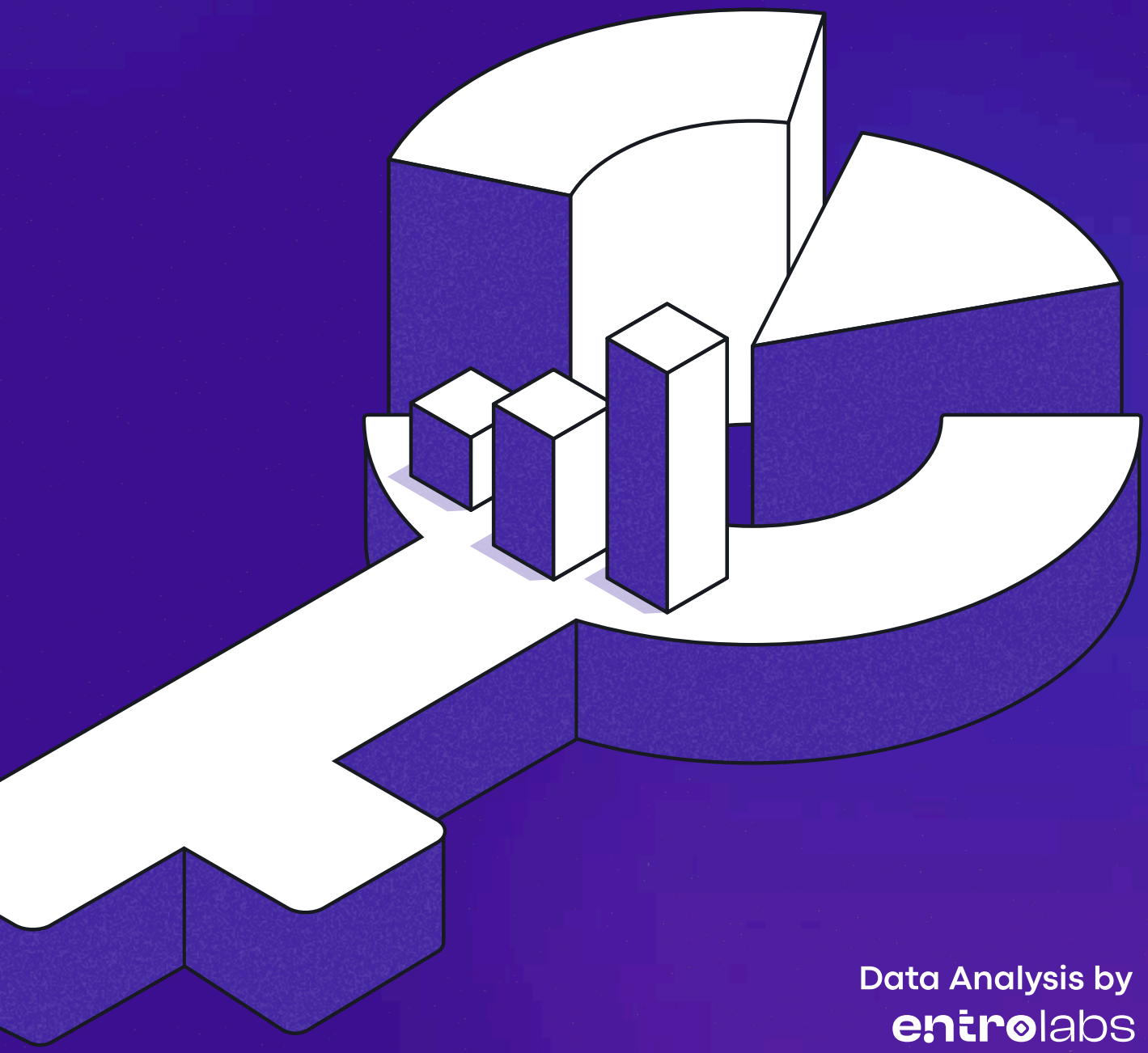
The NHI & Secrets Risk Report H1 2025

Trends, threats and insights from the frontlines of non-human identity & secrets security



Data Analysis by entrolabs

Table of Contents

Executive Summary

Non-human identities (NHIs) are the fastest-growing identity category in enterprise environments. In the first half of 2025, Entro Labs observed a **44% increase** in NHIs (compared to H1 2024), with NHIs now outnumbering human identities by a ratio of **144:1.**

As the number of NHIs grew, so did the sprawl of **secrets**, the credentials they use to authenticate, gain access, and operate. This expanding attack surface is often unmanaged: long-lived keys are left unrotated, secrets are pasted into spreadsheets or shared on messaging apps, and IAM roles with excessive permissions continue to fly under the radar.

This report, built from telemetry across Entro's enterprise customer base, surfaces the most critical risks and threats in NHI and secret security today, from idle yet over-privileged AWS roles, to plaintext secrets buried in collaboration tools. Using hard numbers, the Entro Labs team highlights the structural blind spots in cloud, app and identity security and what security and IAM professionals can do today to close them.

Key Findings

44% Growth in NHIs YoY

The number of non-human identities in the average enterprise increased by 44% between H1 2024 and H1 2025.

144:1 NHI-to-Human Ratio

Non-human identities now outnumber human identities by 144 to 1 a 56% increase from the 92:1 observed in H1 2024.

Shift-Left ≠ Shift-Enough

43 % of all exposed secrets surface outside source code.

Shadow Admins

1 in 20 AWS machine identities carries full-admin privileges ("Super NHIs").

Cloud Sync Hazard

SharePoint holds troves of secrets originating from auto-synced files on endpoints. Half are found within spreadsheets (.xls).

Leaking Tickets

On average, an enterprise has about 1,400 Jira tickets containing plaintext secrets.

Digital Fossils

7.5 % of machine identities live for 5–10 years, outliving both their workload and human owner (1 in 1,000 survives past a decade).

Leaking Pipelines

CI/CD workflows drive 1 in every 4 exposure incidents.

Slack is #1

The most leaked third-party apps credentials, amounting to 42 % of SaaS secrets exposures.

ReposLoaded

One private GitHub repository contains~102 hard-coded secrets on average.

Overprivilege Bloat

8.7 % of NHIs hold permissions they never use.

Top Secret:

Where Secrets Really Leak in 2025

In H1 2025, Entro Labs analyzed hundreds of thousands of real-world secret exposure incidents across enterprise environments and mapped where those leaks actually occurred.

By grouping the findings by type of location, from code and CI/CD workflows to cloud server less functions and collaboration platforms, we uncovered a few surprises.

Unsurprisingly, code remains the most dominant source of secret exposures.

Over 57%

of all exposed secrets came from repositories, commits, or pull requests, reaffirming that "Shift Left" scanning still matters.

But the bigger story is that almost half of all exposed secrets were found outside of code.

The chart below shows the top sources where exposed secrets of all kinds were detected.

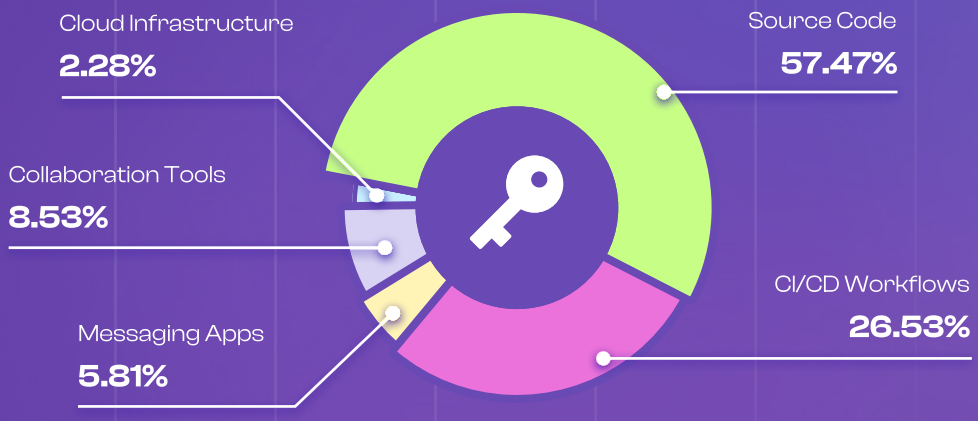
entrolabs insights

- **Code is still king,** but CI/CD pipelines are now a close second, with almost **one in every four** exposed secrets found in DevOps automation logs, build pipelines, or workflow configs.
- **Messaging apps and collaboration platforms are creeping up.** Slack, Microsoft Teams, Jira, Confluence, and SharePoint collectively account for nearly 14% of total secrets exposures. These tools weren't built with secrets in mind, but they're becoming silent repositories of sensitive machine credentials and API keys.
- **Cloud infrastructure isn't exempt.** From environment variables in functions to non-encrypted AWS parameter stores, even small missteps here can create a massive downstream **blast radius,** especially in production environments.

For Security Leaders, The Message Is Clear:

"Secrets Sprawl" Isn't Confined To Code, In 2025 It's Embedded Across Systems, Tools And Workflows.

Detection Must Evolve Accordingly.



Source Code: Where It Usually All Begins

Despite years of secure coding campaigns and "Shift Left" evangelism, **hardcoded secrets remain the #1 cause for secret exposure in organizations**, responsible for almost 60% of all detections in H1 2025.

1/2

of all exposed secrets were found inside version control systems - in repositories, commits and pull requests.

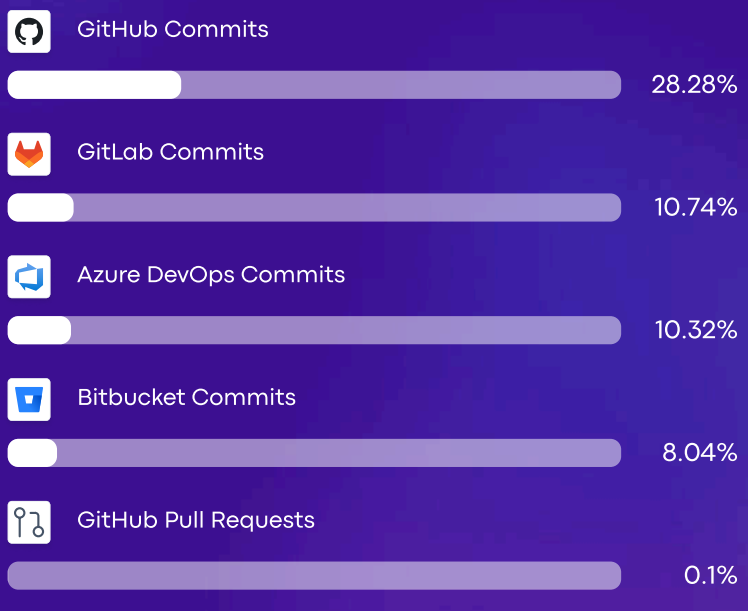
Why? Because code is where developers live. It's fast, easy, and tempting to embed programmatic access credentials directly into scripts and config files, especially in fast-moving teams or MVP environments. But the blast radius of a single leaked token can stretch far beyond its exposure location.

Even with GitHub's built-in secret scanning features, we're seeing sharp growth in plaintext credentials year over year, particularly in **generic secrets** that don't get caught by pattern-based detection.

Here's how it breaks down by platform:

Version Control Platforms

Where code-based secrets leak most



entrolabs insights

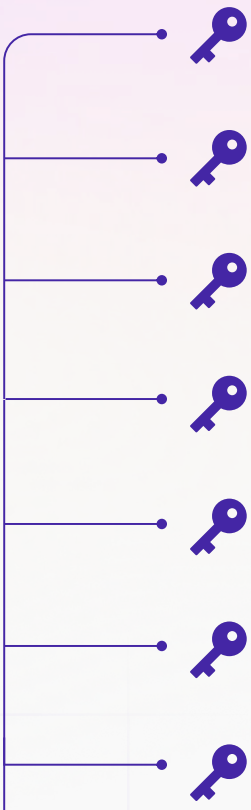
- **GitHub dominates** source-code leaks, with nearly **30%** of all exposures. This tracks with its market share but also raises flags about default config practices and token reuse.
- **GitLab and Azure DevOps** aren't far behind showing that **secrets-in-code is an ecosystem-wide problem.** Bitbucket was also a nontrivial source of leakage, often tied to Atlassian shops and older SDLCs.
- **Pull requests** are rare but high-impact. PRs may expose secrets in config files or .env diffs, often before merging or approval. But even those unmerged PRs retain a full history, including any plaintext secrets that were introduced and later removed from .

GitHub's (Not So) Secret Problem

GitHub, the most widely adopted version control platform, has made notable investments in secrets scanning to reduce accidental exposure of known credential formats. Features like Push Protection have **reduced incidents involving well-defined credentials** (like AWS access keys or Slack tokens). However, GitHub still **falls short when it comes to generic secrets,** such as internal tokens or custom service creds that don't match known patterns (also known as regex, regular expressions).

And while GitHub's native and premium secrets scanning tools are powerful, they're not a silver bullet. Their effectiveness depends heavily on:

- Developer behavior and enforcement of secure coding practices
- Proper configuration of scanning and alerting features
- Integration into a broader NHI lifecycle and secrets security strategy that goes beyond code - spanning CI/CD, SaaS apps, cloud infrastructure etc.



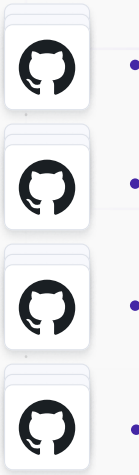
102

EXPOSED SECRETS PER GITHUB REPOSITORY

We analyzed data from **13,841 private GitHub repositories** across Entro's customer base.

On average, each contained a staggering **101.83 exposed secrets,** API keys, tokens, certificates and other credentials hardcoded.

Each one represents a potential foothold for attackers, significantly expanding risk across development environments.



What Security Leaders Should Do Now

Because many of your leaked secrets originate beyond code repos, mandate organization-wide secret scanning projects for logs, chats, and file shares – and cap log retention at 90 days to limit exposure.

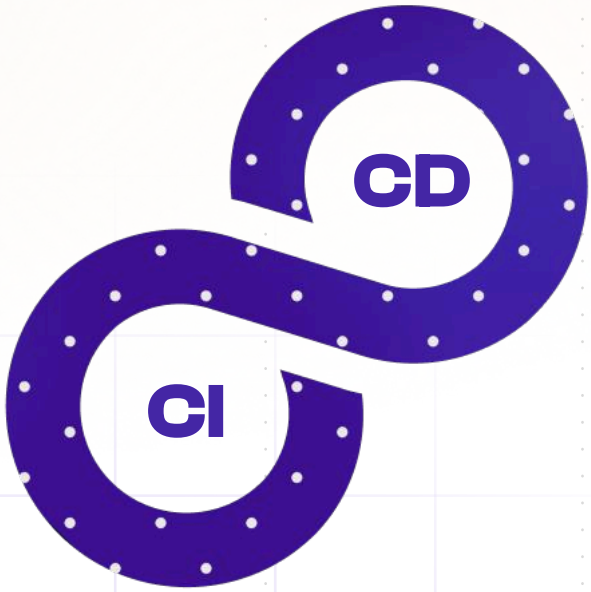
CI/CD Workflows: Automation That Leaks Secrets

CI/CD tools automate builds, tests, and deployments. They are speeding up development and ensuring rapid software delivery. But this automation brings an often-overlooked risk: secret exposure in pipeline/workflow logs. These logs frequently capture environment variables, tokens, and API keys that get unintentionally output during debugging or misconfiguration.

In H1 2025, **CI/CD workflows accounted for over 26% of total secret exposures, second only to source code itself**. GitHub Actions alone represented 24.75% of these exposures, alongside tools like Buildkite, Jenkins, GitLab CI, and CircleCI, which face similar risks.

Developers and DevOps teams often overlook that secrets logged during pipeline execution remain persistent and accessible to anyone with sufficient permissions, making them high-value targets for attackers.

Developers and DevOps teams often overlook that secrets logged during pipeline execution remain persistent and accessible to anyone with sufficient permissions, making them high-value targets for attackers.



The TJ-Actions Supply Chain Attack

A high-profile example emerged in March 2025, when attackers compromised the popular tj-actions GitHub Action using a stolen GitHub **Personal Access Token (PAT)** - a non-human identity. Malicious code was stealthily inserted into the action source code, propagating across more than 23,000 repositories.

This code silently captured and exfiltrated pipeline-logged secrets of many organizations worldwide. The vulnerability first detected by StepSecurity was later revealed to be part of a highly targeted supply chain attack against Coinbase, a major cryptocurrency exchange.

This incident highlights two of the major risks this report covers, exposed secrets and compromised NHIs showing how attackers leverage NHIs to amplify credential leaks at scale.

Messaging & Collaboration Tools: Send. Share. Expose

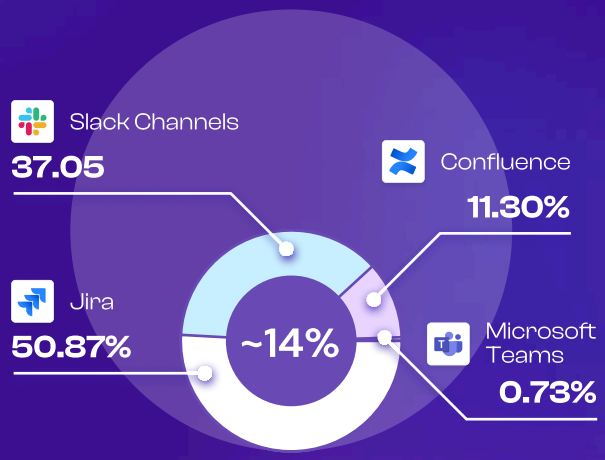
Collaboration platforms and messaging apps are core to fast-moving, SaaS-first organizations - they're also quietly becoming hotspots where end users expose secrets in 2025.

Developers and engineers often paste snippets of code, API keys and other types of credentials into Slack channels, tickets on ServiceNow, and internal wikis for convenience and as part of rapid troubleshooting, onboarding or integration processes. Once posted or shared via a message, a secret remains accessible indefinitely, visible to other users within the org (in best cases where the channel/team is private) - or potentially to attackers who gain account access.

In H1 2025, **Messaging and Collaboration Platforms were the source for ~14% of exposed secrets** (excluding SharePoint we'll get to that*). This risk surface is growing steadily as remote work and distributed teams rely increasingly on fast communication and collaborative frameworks.

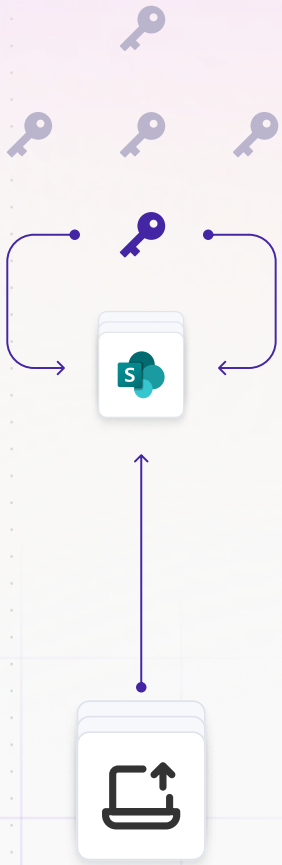
Our analysis found an average of **1,415 Jira tickets per organization contained plaintext secrets.** Whether it's API keys dropped in bug reports or tokens shared for debugging, ticketing systems like Jira and ServiceNow are often overlooked vectors for secrets sprawl, making ITSM platforms silent risk amplifiers in the enterprise.

Share of Secret Exposures by Collaboration Platform



Learn more about secret exposure in ServiceNow: Leaking Tickets

Spotlight: SharePoint and the Goldmine of Secrets

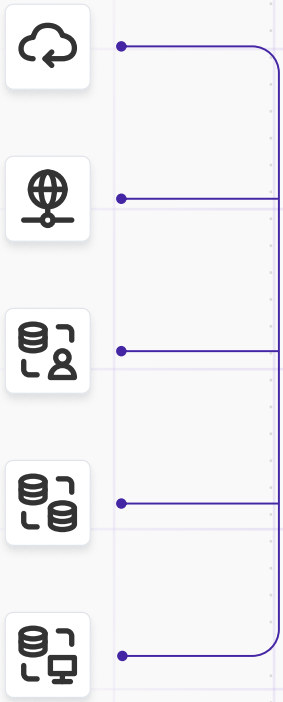


When collecting the data for this report, SharePoint alone accounted for **almost 20% or 1 in every 5 exposed secrets**, a major outlier we chose to exclude from initial totals for the primary analysis.

The reason is technical but significant for security: in Microsoft shop enterprises, **SharePoint often syncs automatically to end-users local devices via OneDrive sync**. OneDrive is often configured by default to automatically sync or "back up" files from Desktop, Documents, and Pictures folders when setting up Windows 11.

This means files containing secrets (or not) in local folders (e.g., Desktop) are automatically uploaded and stored in the cloud in those files, making plaintext secrets accessible to the org's M365 admins across the endpoints and locally via any account that syncs its files.

In essence, **SharePoint acts as a quiet cloud aggregator of secrets** and most Microsoft 365 users aren't even aware.



"The secret is in a local file I saved on my laptop, what harm could it possibly do?"

Usually, not much, unless the machine itself gets compromised (say, by an info stealer). But SharePoint/OneDrive sync complicates things:

- If a user's account is breached, synced files from their local machine - codebases, configuration files, scripts - can be accessed via SharePoint on the cloud, including any embedded secrets they may include.
- If an M365 admin account is compromised, the attacker may gain access to secrets synced across the entire organization's SharePoint instance.

Files sync automatically between local endpoints and SharePoint, creating a hidden secrets repository in the cloud.

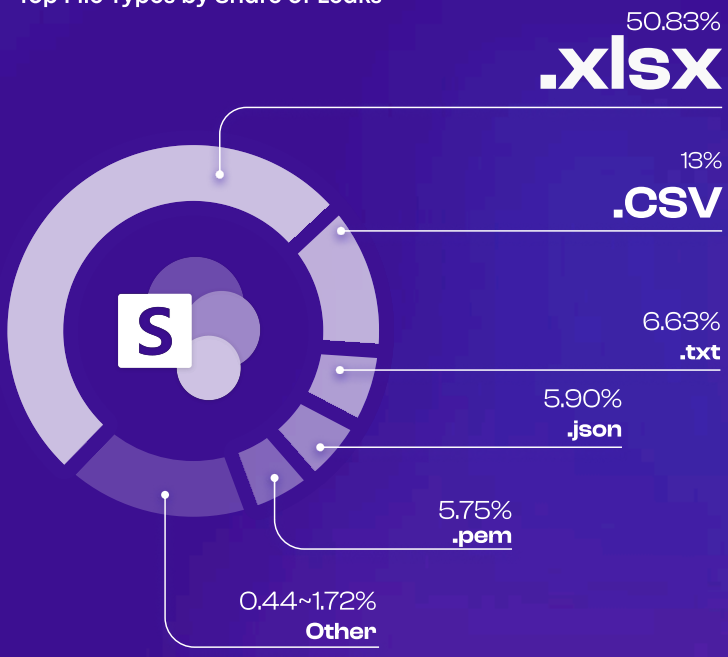
Which SharePoint-Hosted Files Leak The Most?

We analyzed every SharePoint-related secrets finding from the past six months to identify the file types most prone to contain plaintext secrets in SharePoint instances. In almost every case, the culprits fall into two buckets:

1. Locally-saved files that sync via OneDrive backup
2. Files users actively share through Microsoft 365 collaboration apps

Where SharePoint Secret Hide

Top File Types by Share of Leaks



Why this matters: Office and data files move with almost no friction, shared, edited and re-shared by many hands, so when they sync to SharePoint they suddenly inherit broad, cloud-level access. Yet most secret-scanners ignore these formats, concentrating instead on code and CI/CD artifacts, spreadsheets and note files slip straight past the scanners.

The result is a huge blast radius: a single M365 admin, or any over-privileged compromised user account, can easily access every synced secret across the tenant.

entrolabs insights

- **Spreadsheets are the #1 danger zone.** Over half of all SharePoint-hosted secrets were inside **.xlsx** workbooks, logs and "tracking sheets" that developers paste credentials into for convenience.
- **Data dumps come in second**. CSV files add another **13%.** Together, XLSX + CSV account for over **60% of exposed secrets** we found in SharePoint, yet these formats rarely run through secret-scanning procedures.
- **Plain-text is alive and well.** TXT, JSON, and PEM files collectively contribute another **18%.** These are quick-and-dirty notes, config fragments, or certificate bundles that get synced to the cloud by default.
- **Scripts and docs aren't innocent either.** PowerShell **(.ps1),** SQL dumps **(.sql),** Word docs **(.docx),** and even OneNote **(.one)** each surface credentials, showing **that any file type can become an unsecure secret "vault".**

What Security Leaders Should Do Now

Treat SharePoint files with the same scrutiny you give source code. Expand secret-scanning to office formats, restrict who can read synced folders, and remind teams that "just a spreadsheet" can contain the literal keys to the kingdom.

Top 25 Most Exposed SaaS Secrets

So we've seen where they get exposed, now let's discuss which secret types get exposed the most. When secrets leak, they're rarely abstract. They're tied to services, tools and apps that organizations rely on every day. In our analysis we identified the 25 most commonly detected SaaS-related secret types across the stack.

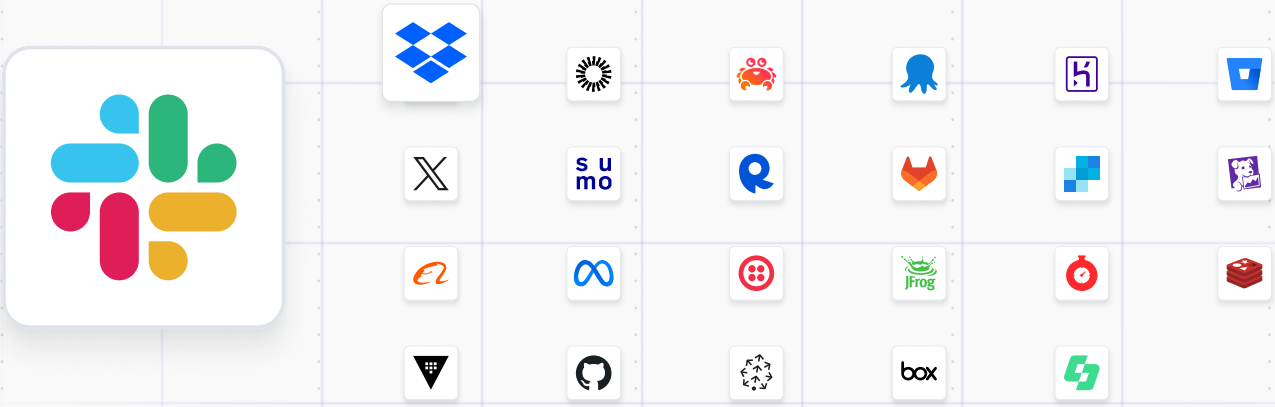
Slack tokens topped the list (by a great margin), accounting for **over 40% of all SaaS secret exposures**. As a ubiquitous messaging app deeply integrated into developer workflows, Slack is often wired into CI/CD pipelines, security/alerting systems, bots, and internal tools. That convenience makes its tokens easy to generate and unfortunately, just as easy to expose.

Other frequently exposed secrets include API keys for **Dropbox** (used for programmatic file access), **GitHub**, **Pinecone**, **Heroku** and **Bitbucket**, all common in modern development stacks.

Notably, we also found API keys for platforms like **Twilio**, **SendGrid** and **Meta**, used for messaging, email delivery and advertising automation. These services typically rely on long-lived API keys, and a single exposed key could allow attackers to send messages, drain budgets or harvest sensitive metadata.

The list includes everything from infrastructure tokens (e.g., Redis, HashiCorp, Datadog) to social integrations (Facebook, X/Twitter), demonstrating how deeply third-party apps are embedded in the enterprise stack in 2025.

If your SaaS tools issue an API key, chances are it's been exposed somewhere. The attack surface isn't just in your code or your cloud infra - it's in the ecosystem of services around it.



What Security Leaders Should Do Now

With Slack bot tokens alone driving more than 40% of SaaS-secret leaks, immediately inventory every third-party API key, beginning with Slack, and force-rotate or delete any long-lived or over-scoped tokens; mandate the usage of vaults, then wire continuous secret-scanning into chat, CI logs and cloud storage and impose short, automatic expiry windows with alerts for any token spotted outside approved secret management solutions.

NHI Boom:

Agentic AI Accelerates Identity Sprawl

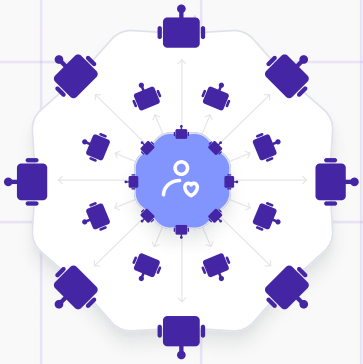
Between H1 2024 and H1 2025, the number of non-human identities in the average enterprise

Grew by 44%

Based on prevailing industry trends and observed enterprise behavior, Entro Labs attributes this growth to the rapid adoption of agentic AI and automation-first development practices. As organizations integrate large language models (LLMs), deploy autonomous agents and expand API-driven workflows, each new agent requires more service accounts, access tokens and machine credentials into the environment. Most of them are created automatically, few are governed.

New Ratio, Bigger Risk: The ratio of **NHIs to human identities** in the enterprise has jumped from 92:1 (H1 2024) to 144:1 in H1 2025, a 56% increase in just a year.

92:1 H1 2024



144:1 H1 2025

This shift reflects a structural change in enterprise architecture. Apps now talk to apps more than humans talk to apps. And that means attackers will, too. Each new NHI is a potential entry point, especially if secrets are exposed, ownership is unclear or privileges are excessive.

What Security Leaders Should Do Now

Traditional IAM practices designed for humans no longer scale in 2025. Organizations must adopt NHI-first visibility, governance and risk detection models.

NHIs & Secrets:
Aging Into Risk

Much like our employee users, the non-human identities and secrets in the organization have a distinct lifecycle: **creation**, **usage**, **rotation** and ultimately, **retirement** or **decommission**. Simple enough, yet in practice many organizations struggle, letting their machine and workload identities age dangerously.

In our previous Entro Labs publication on LLMJacking, we observed that once AWS credentials are exposed, threat actors begin probing them in under 17 minutes on average, and in some cases, the time to attack is as fast as 9 minutes, to assess access to GenAI services and other sensitive resources.

This narrow window speaks to the importance of NHI lifecycle management and regular secret rotation. If credentials are short-lived or auto-expire, the blast radius of a leak is dramatically reduced. But our data shows that's rarely the case.

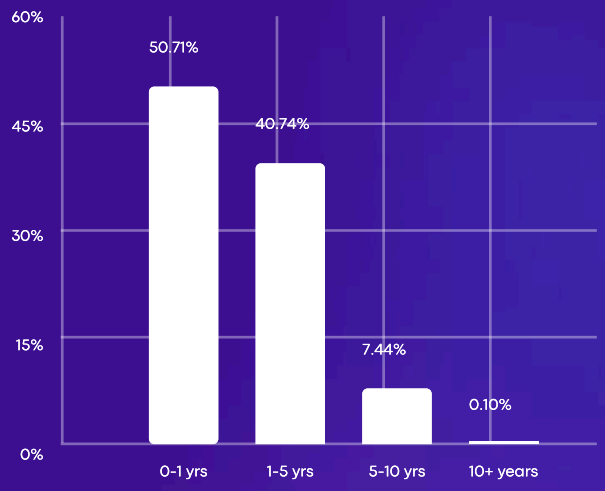
What we found next highlights just how stale many of these credentials have become.

NHIs Built to Last, Forgotten Fast

We found that **nearly half of all active NHIs are over a year old**, with 7.5% aging between 5–10 years. These identities often outlive their intended use and their owner, remaining active without rotation, visibility or ownership. As they age, they become prime targets, particularly in the critical window after exposure. Without lifecycle management, stale NHIs quietly expand the attack surface.

NHI **Age Distribution**

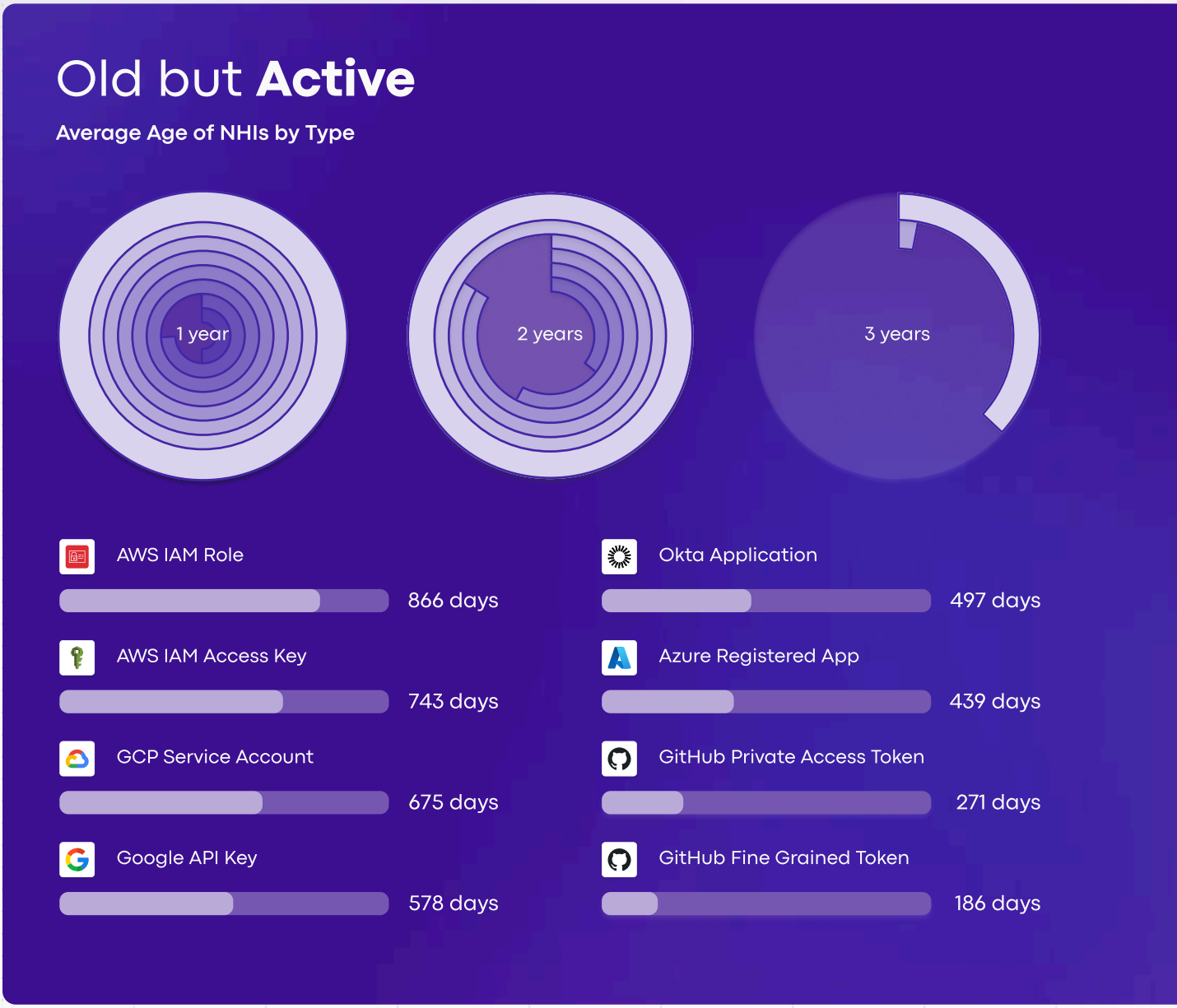
How Long Do Machine Identities Stick Around?



1 out of every 1,000 NHIs is over 10 years old.

In contrast, **the median employee tenure**, according to the U.S. Bureau of Labor Statistics **is just 3.9 years** : the median human user stays with their employer just 3.9 years. While human users naturally cycle through the organization, **many NHIs quietly outlive their creators**, persisting with privileges long after they're forgotten.

The Oldest NHIs in the Stack



Some of the most aged and persistent non-human identities in enterprise environments are also the most impactful:

- AWS IAM Roles average nearly **2.4 years**
- AWS Access Keys sit at **2 years**
- GCP Service Accounts: **1.85 years**
- Google API Keys: **1.6 years**
- Okta Applications: **1.36 years**
- Azure Registered Apps: **1.2 years**

Even developer-side tokens show surprising "longevity":

- GitHub Personal Access Tokens: **9 months**
- GitHub Fine-Grained Tokens: **6 months**

AWS NHIs: Most Are Sitting Ducks

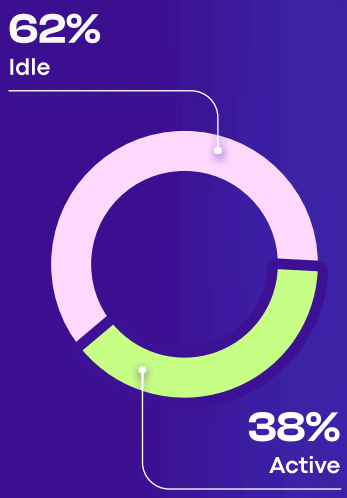
In AWS environments we have analyzed, **over 62% of non-human identities showed no activity in the 90 days** leading up to data collection time.

These idle NHIs tokens, IAM users and access keys represent a massive pool of forgotten machine identities that attackers can silently abuse if exposed.

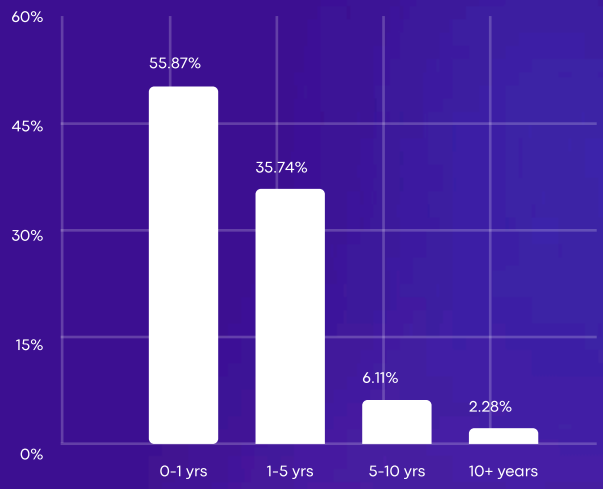
Worse, many of these idle identities still retain permissions over AWS services, some even with administrative privileges (as we'll see in the next section).

Idle vs Active

Based on activity in the last 90 days



Secrets Age Distribution



Secrets Don't Age Well

Secrets show a similar aging trend to NHIs, only worse. While over 55% are under a year old, a surprising **2.3% of all active secrets are over 10 years old**, more than 20x the share of decade-old NHIs. Many of these aging secrets are perceived by R&D teams as foundational to the software they support, untouchable, deeply embedded and too complex to replace.

As a result, they're rarely tracked or reviewed and without auto-expiry or rotation policies, they slowly accumulate in code, configs and vaults.

Overprivileged NHIs:

Blind Spots in IAM Governance

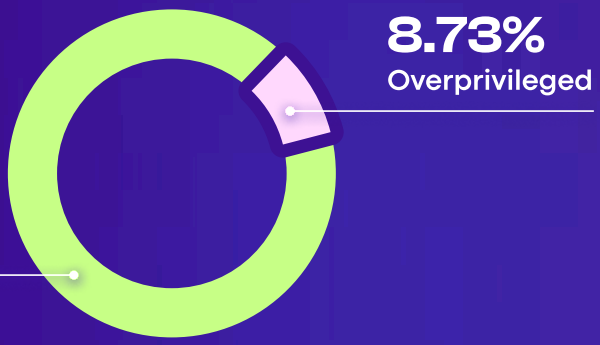
When NHIs are created without clear ownership or scoped permissions, they don't just live forever, they expand unchecked. This often results in tokens and IAM roles that touch far more services than they should, without anyone noticing. According to IBM Security, IAM teams only govern ~44% of machine identities in the enterprise. The rest, service accounts, API tokens, access keys left unmanaged.

In our analysis of AWS environments, we examined how well-scoped non-human identities actually are. We defined **Overprivileged NHIs** as those scoped with permissions that exceed their actual usage patterns.

These are machine identities granted access to services and actions they rarely or never interact with in practice.

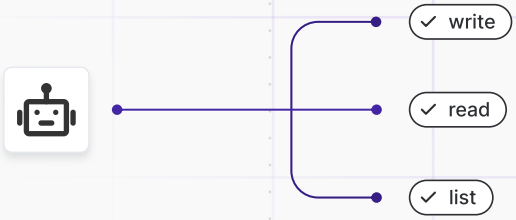
Overprivileged vs Right Sized

NHI Posture Risks in AWS



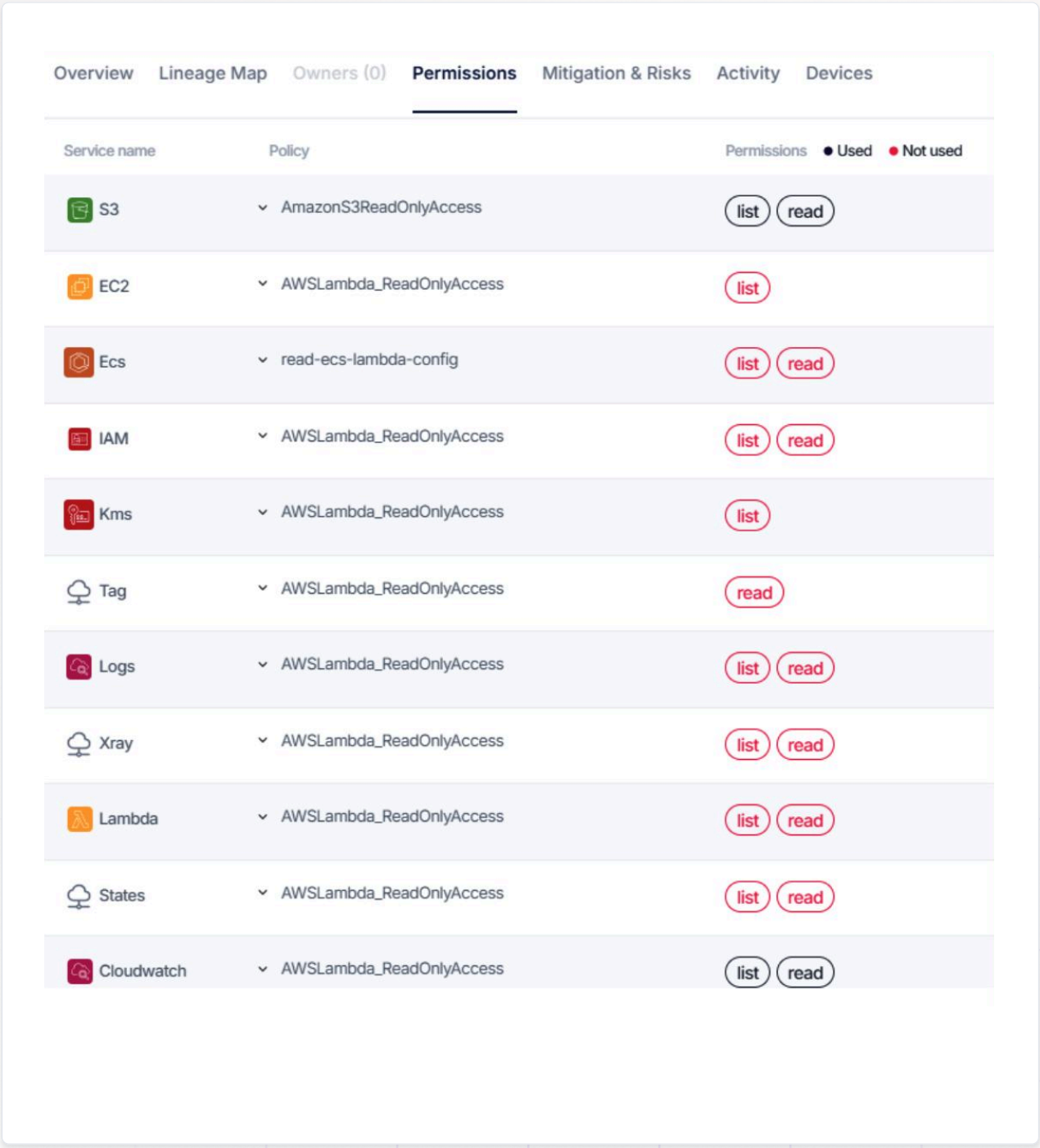
This data reflects Entro's customer base, organizations actively working to reduce NHI posture risk by right-sizing and decommissioning idle ones, meaning the real-world number is likely much higher in less mature environments.

Almost 9% of NHIs showed no activity across some of their granted permissions, making them both excessive and idle.



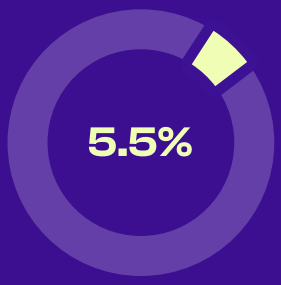
This over-permissioning isn't just about inactivity, it's about unnecessary reach that expands the blast radius of each NHI.

On average, a single AWS IAM role can access 43 different AWS services like **S3 buckets, EC2 instances, Lambda functions, RDS databases and Secrets Manager**, far beyond the scope needed for most workloads.



Super NHIs: AWS Admins in the Shadows

We analyzed AWS environments to identify how many non-human identities hold admin-level permissions. We found a staggering number, on average, **over 5.5% of all AWS NHIs are "Super NHIs"** machine identities and roles with administrator access across services.



Over 5.5% of all AWS NHIs are Super NHIs

*Super NHIs: a machine identity with admin permissions

This percentage ranged significantly across organizations:

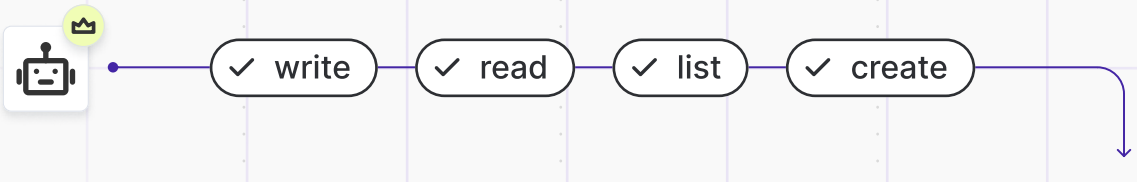
- Some tenants had as low as **0.15%,** reflecting strong privilege hygiene.
- Others had **nearly 18%** of their AWS NHIs configured as full admins.

These Super NHIs, a term inspired by Microsoft's concept of "super identities", are non-human actors with elevated privileges that in most cases far exceed their operational necessity. While they may enable automation and integration, their excessive power makes them critical risk multipliers. If compromised, they offer attackers unrestricted access to cloud infrastructure and sensitive data.

1 in 20

AWS machine identities is a full admin

Super NHIs hold elevated privileges across services (often unnecessarily) turning them into prime targets with the widest blast-radius.

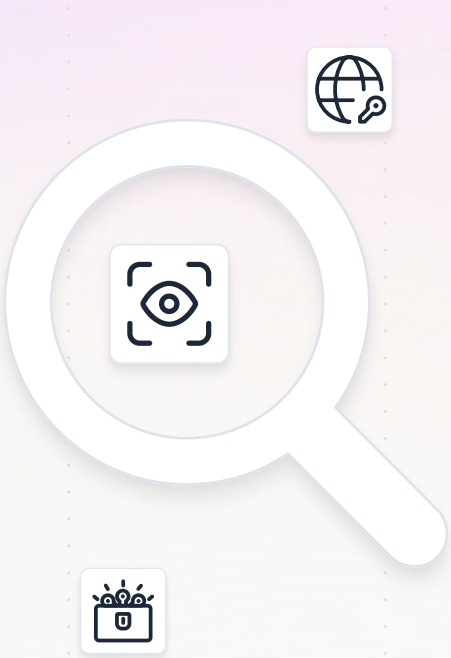


What Security Leaders Should Do Now

Because many of your AWS machine identities carry "*" level or AdministratorAccess policies, start with a ruthless sweep, locate and delete any IAM roles that are unused or unnecessary, then quarantine the few essential admin NHIs in a locked-down account, enforce MFA or hardware keys and audit every AssumeRole call.

NHIDR™ Risk Radar:

Top 5 NHI Anomalies Detected in H1 2025



Entro's **NHIDR™** engine (Non-Human Identity Detection & Response) continuously monitors the behavior of tokens, service accounts and other NHIs across enterprise environments. By correlating usage patterns, access logs and environmental signals in real-time, NHIDR™ flags abnormal activity that suggests compromise, misuse or policy violations.

In the past six months, these were the five most common risks across our customer base:

1.

Token Used by Multiple Devices:

An NHI credential that is accessed from multiple machines or applications indicates possible sharing, duplication or leakage. In legitimate usage, most service tokens are tied to a specific workload so broad usage often suggests compromise or misconfiguration.

2.

Previously Inactive NHI Becomes Active Again

A long-dormant token suddenly reactivating can indicate reconnaissance attempts or an ongoing attack. This often means an old credential, thought to be unused or forgotten, has resurfaced - possibly found and exploited by an attacker.

3.

Secret Accessed by a Human

Secrets fetched manually via browser, CLI, or IDE are an immediate red flag. This behavior often points to credential hunting, insider misuse or lateral movement attempts.

4.

Activity from a Restricted IP

When a NHI that is usually being accessed from known IPs (e.g., corporate office, VPC) is used from an unknown or geofenced location, it likely signals unauthorized access or token exfiltration.

5.

Previously Disabled Token Reactivated

If an NHI token was deactivated but suddenly comes back to life, it can signal either misconfiguration or intentional reactivation by an attacker.

Methodology

The findings presented in this report are based on proprietary data collected and analyzed by Entro Labs across enterprise environments between **January 1 and June 30, 2025**. The analysis encompasses real-world non-human identities and secrets telemetry from global customers operating in diverse industries and cloud architectures.

Data Sources

- NHIs and Secrets Inventory: Automatically discovered across cloud providers, source code, CI/CD pipelines, collaboration tools and SaaS platforms using Entro's agentless integrations and API or on-prem connectors.
- Usage and Permissions Analysis: Entro correlated access logs, IAM policies, and secret usage signals to assess behavior patterns, permission sprawl and privilege utilization.
- Exposure Detection: Secrets exposure incidents were identified via Entro's detection engine, including findings from code repositories, workflows, SharePoint instances, Jira tickets, messaging apps, and misconfigured cloud environments.

Limitations

While the customer base represents mature security-conscious organizations actively reducing NHI and secret risks, we suspect that many of the reported statistics (e.g., idle NHIs, admin privileges) may underrepresent broader industry baselines. Real-world figures in less mature environments are likely higher.

Scope

- Includes hundreds of thousands of exposed secrets and over 27 million NHIs.
- Covers AWS, Azure, GCP and leading SaaS ecosystems (e.g., GitHub, Slack, Jira, ServiceNow, SharePoint).
- All data is anonymized and aggregated, no customer-identifiable information is included.

Key Definitions

- Secrets Exposure: Any instance where a credential (e.g., API key, token, certificate) was found outside a secure location in plaintext.
- Overprivileged NHI: A machine identity with granted permissions exceeding its recent usage patterns.
- Idle NHI: An NHI that exhibited no usage activity across any granted permission over a 90-day window.
- Admin NHI ("Super NHI"): Any NHI with administrator-level or equivalent permissions in cloud environments.

About Entro

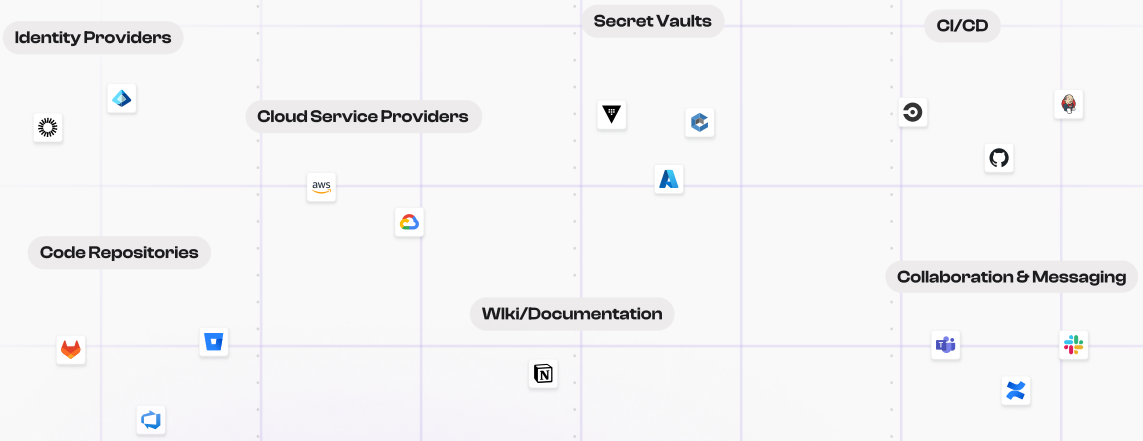
Entro is the pioneer in Non-Human Identity (NHI) and Secrets Security, helping enterprises discover, manage, and secure machine identities across cloud, code, CI/CD and collaboration environments.

The Entro platform builds a complete, contextualized inventory of tokens, service accounts, access keys, and secrets across the enterprise stack. It maps usage, assigns ownership, enforces best practices and detects suspicious behaviors in real time via the proprietary NHIDR™ (Non-Human Identity Detection & Response) capabilities.

Entro provides full NHI lifecycle management - from creation to rotation and includes ContextIQ™, an AI-powered triage engine that filters false positives and enriches exposure insights to reduce alert fatigue and accelerate response.

entrolabs

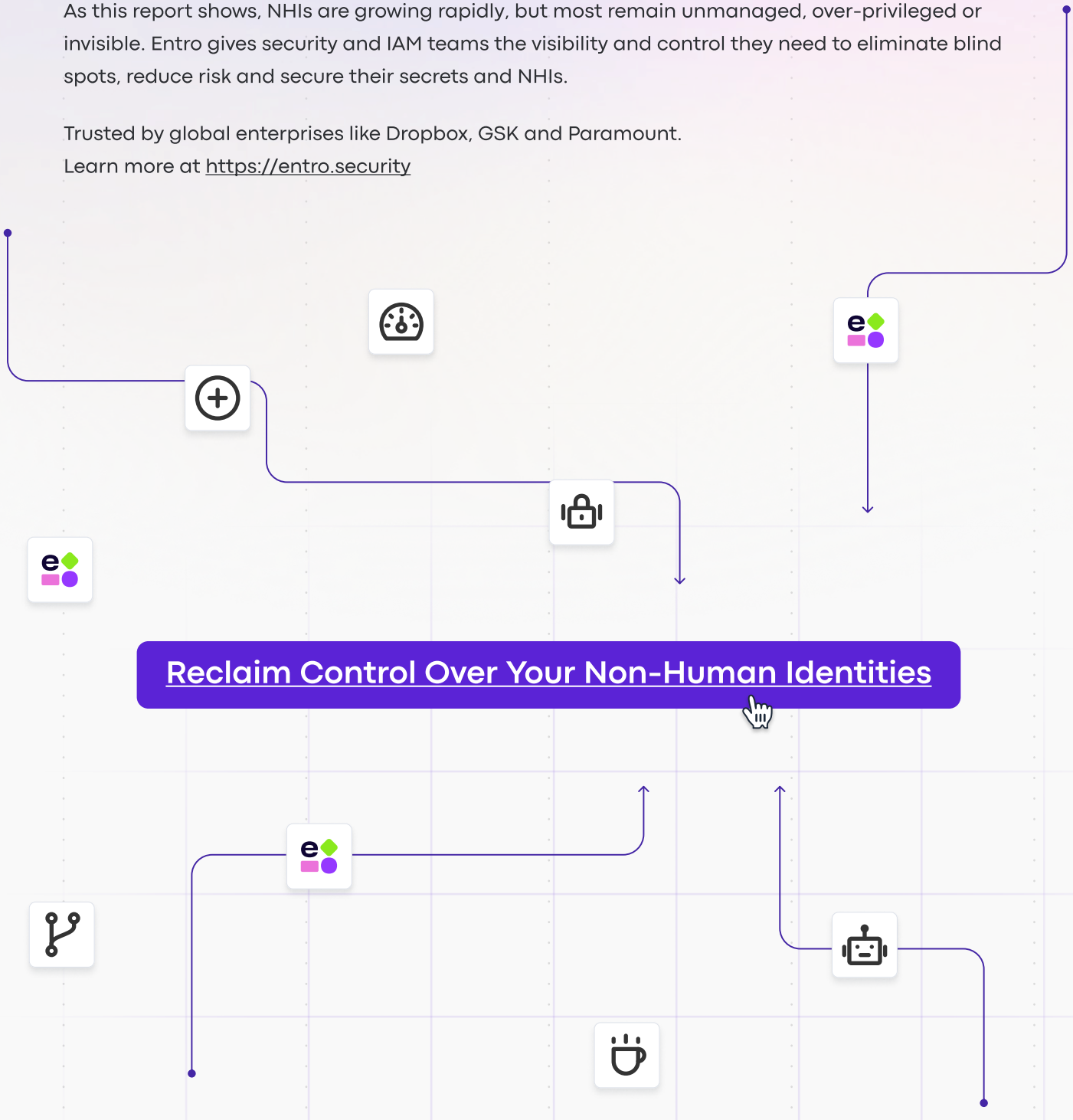
This report was created by Entro Labs, a dedicated team of security researchers and identity experts inside Entro Security. Their mission is to investigate how NHIs are really used, and misused, in the wild, uncovering blind spots, risks, and real-world attacker pathways. Entro Labs continuously monitors trends, misconfigurations and exposures to provide deeper visibility into the NHI threat landscape.



- Discovery & Inventory
- NHI Posture Management
- NHIDR™
- NHI Lifecycle Management

As this report shows, NHIs are growing rapidly, but most remain unmanaged, over-privileged or invisible. Entro gives security and IAM teams the visibility and control they need to eliminate blind spots, reduce risk and secure their secrets and NHIs.

Trusted by global enterprises like Dropbox, GSK and Paramount.
Learn more at https://entro.security



The NHI & Secrets Risk report H1 2025

Data Analysis by entrolabs