GitGuardian

THE STATE OF SECRETS SPRAWL

2022

3.4k

Occurrences of secrets detected per AppSec engineer in 2021

The growing problem of secrets sprawling in corporate repositories can only be solved by enabling **collaboration between AppSec and Developers**.

It's safe to say that 2021 will go down in history for cybersecurity experts.

Ransomware and other large-scale cyberattacks (SolarWinds, Colonial Pipelines) or vulnerabilities (Log4Shell) have made headlines around the world. Software supply chain attacks have seen their number explode, and this comes as no surprise considering the plethora of vulnerabilities and misconfigurations found across software development environments.

Unsurprisingly, a lot of attacks start with the compromise of a leaked secret. Credentials are a nightmare for security engineers because they can end up in so many places: build, monitoring, or runtime logs, stack traces, and ... git history. Our data show the extent of publicly exposed secrets on GitHub **has more than doubled since 2020**. The problem is not bound to this particular platform, as revealed by our Docker Hub analysis.

In 2020, GitGuardian started monitoring private repositories as well, which granted us a unique insight into what really happens behind the scenes.

The data reveals that on average, in 2021, a typical company with 400 developers would discover 1,050 unique secrets leaked upon scanning its repositories and commits. With each secret detected in 13 different places on average, the amount of work required for remediation far exceeds current AppSec capabilities: with a security-to-developers ratio of 1:100*, **1 AppSec engineer needs to handle 3,413 secrets occurrences on average**.

This comforted our view that the only way to address the challenge of secrets sprawling within corporate repositories is to enable a shared responsibility between AppSec and Devs.

* From TAG Cyber, see Methodology

Secret

A secret can be any sensitive data that we want to keep private. When discussing secrets in the context of software development, we refer to digital authentication credentials that grant access to services, systems, and data. These are most commonly API keys, usernames and passwords, or security certificates.

Secret incident

A secret incident is a uniquely identified security event that has been determined to have an impact on the organization and necessitates remediation. An incident often has multiple **occurrences** across files or repositories.

Secret occurrences

A single incident generally encompasses multiple occurrences, which are the various locations across files or repositories where the secret was identified. Occurrences map to the magnitude of the sprawl, and are therefore correlated to the amount of work needed to redistribute the secret after it has been rotated. Occurrences can be assimilated to technical debt.

Secrets sprawl

Unlike traditional credentials, secrets are meant to be distributed to developers, applications, and infrastructure systems. Adding more of these factors will inevitably make the number of secrets used in a development cycle increase, leading to a natural sprawling phenomenon: secrets start to appear hardcoded in source code. From an organization's point of view, visibility and control over their distribution start to degrade. This is what secrets sprawl is all about.

> “ Source code is a huge wealth of knowledge. It also happens to exist on pretty much every developer’s workstation, which they probably take home with them. You probably don’t want your secrets being all over the country.

Don, Security engineer

Public Monitoring

ON GITHUB

56M users

+25% repositories created last year

*from the state of the Octoverse 2021

+23%
commits scanned by GitGuardian

How leaky was 2021?

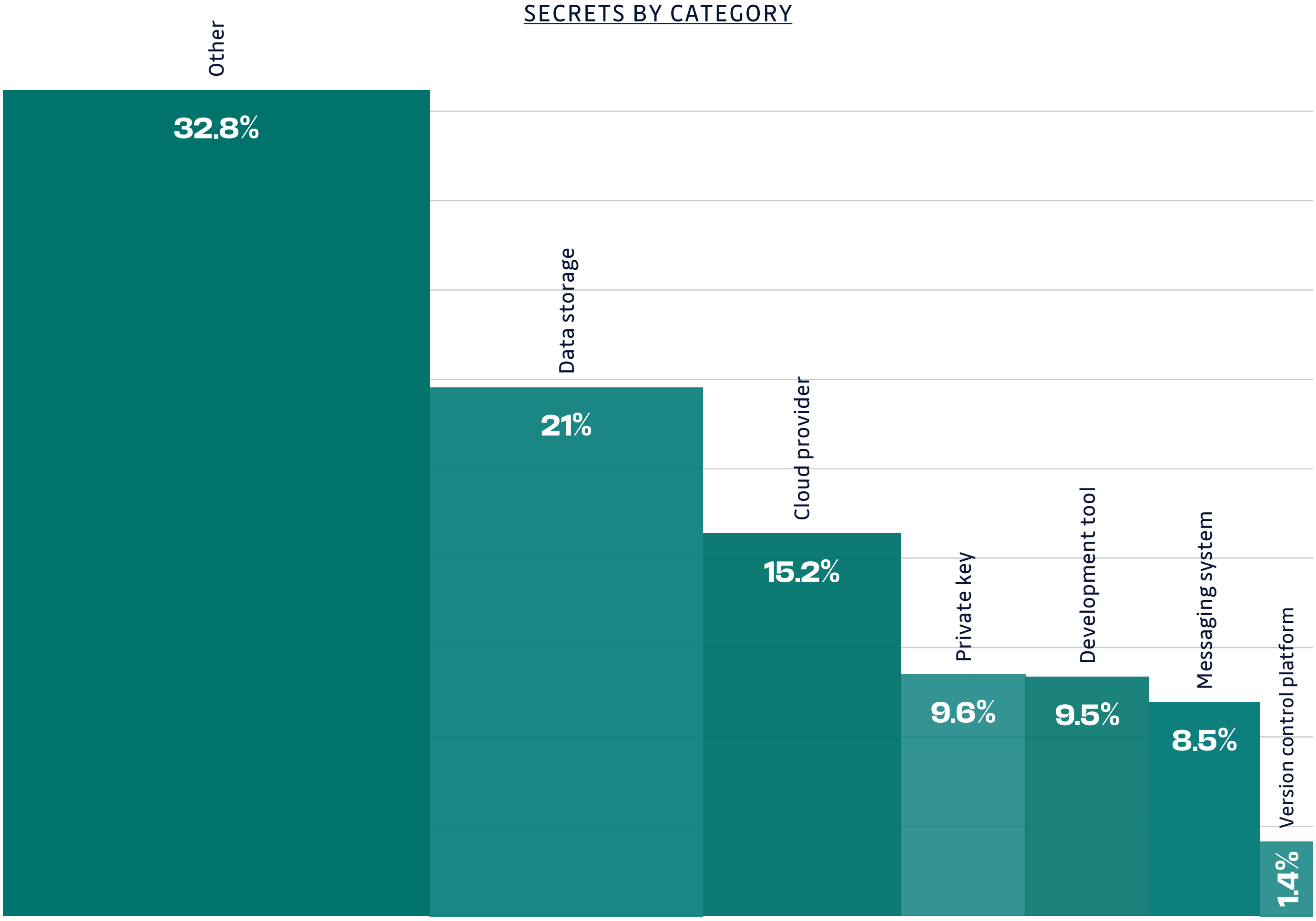
Over **6M secrets** detected in 2021

2X increase compared to 2020

On average, 3 commits out of 1,000 exposed at least one secret,
+50 % compared to 2020*

* see Methodology

SECRETS BY CATEGORY

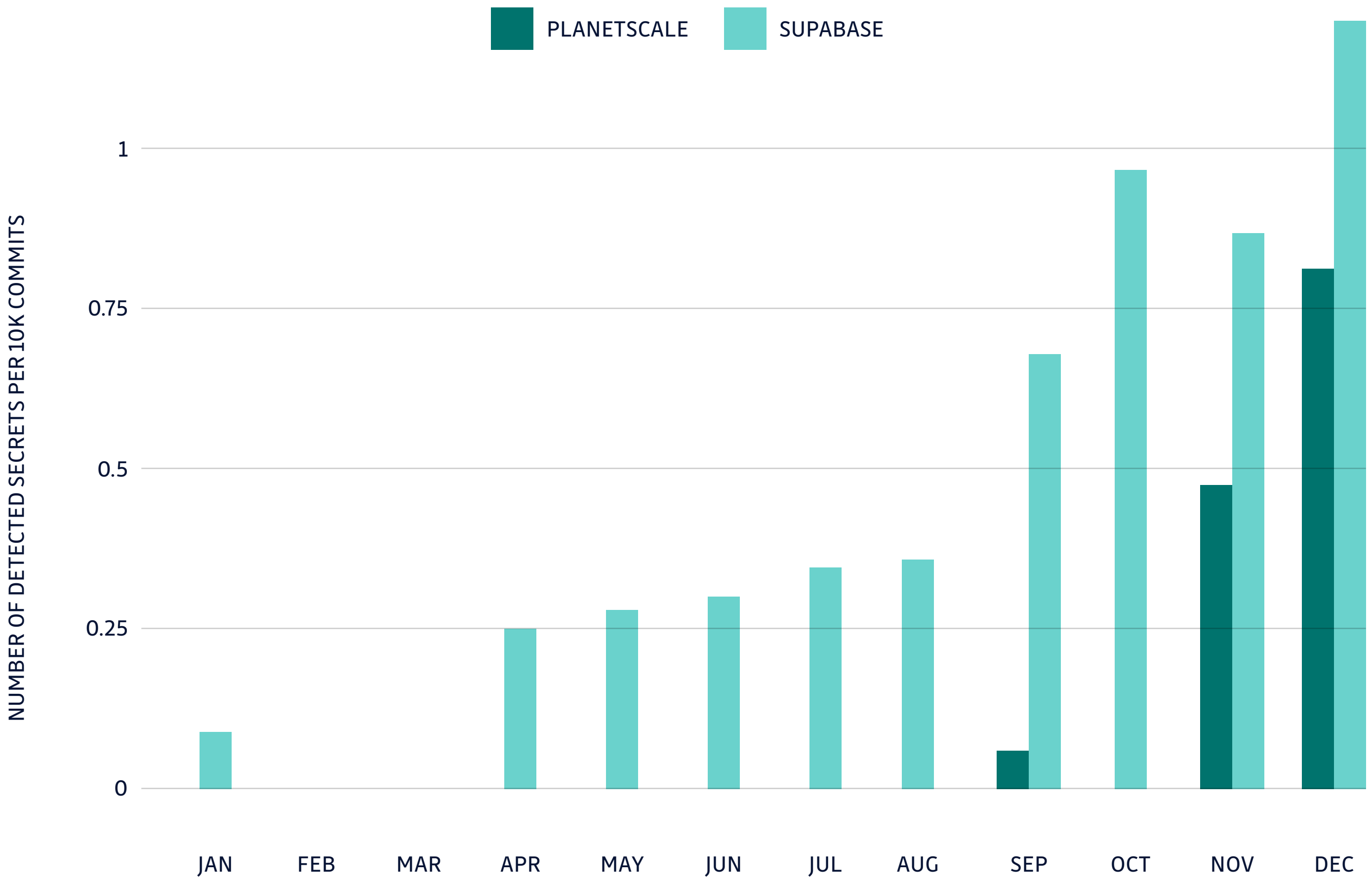


Leaks correlate to popularity

It should come as no surprise that leaks are proportional to user adoption, and this is especially true for newcomers rapidly gaining in popularity.

Supabase, which has consistently ranked in GitHub's top-20 fastest-growing open-source startups and launched 50,000 databases in 2021 only, is a telling example. Another one is **PlanetScale**, a serverless database platform released in 2021 Q4, which immediately started appearing on our radars.

EVOLUTION OF THE NUMBER OF DETECTED SECRETS IN 2021



Scanning Docker Hub

When it comes to open-source, GitHub certainly is the first platform that comes to mind. Yet it is not the only resource for code-sharing. Since Docker popularized the use of containers to package apps, its official public registry, Docker Hub, has become another developers' favorite.

> It's therefore not surprising to find secrets in Docker Hub.

The layers making up Docker images are as many additional attack surfaces that can too easily be left out of the security perimeter. For attackers, it is yet another chance of finding an access vector, just as demonstrated by the Codecov breach.

> Docker Hub = 8.8M Docker images publicly available

This motivated us to conduct our first study on the extent of secrets sprawl in Docker Hub a few months ago. To deepen this first estimation, we reiterated the exercise, this time with a 5x larger sample. Here are our results:

4.62%
of the images expose at least one secret

10K
images scanned

→

4K
secrets

1.2K
unique

6
secrets

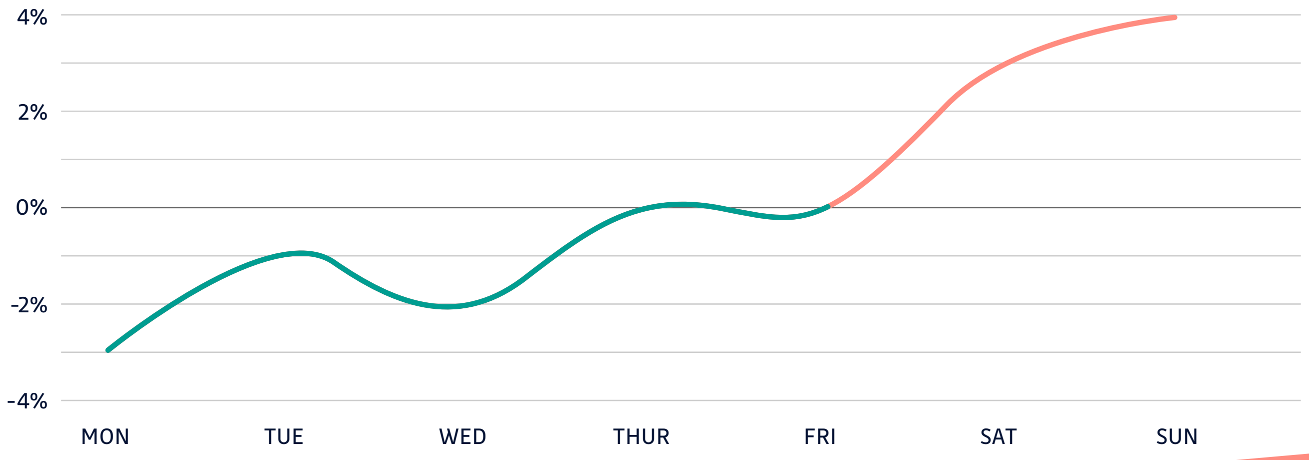
EVERY

100
layers

Fun facts!

Leaks mostly happen
on weekends...

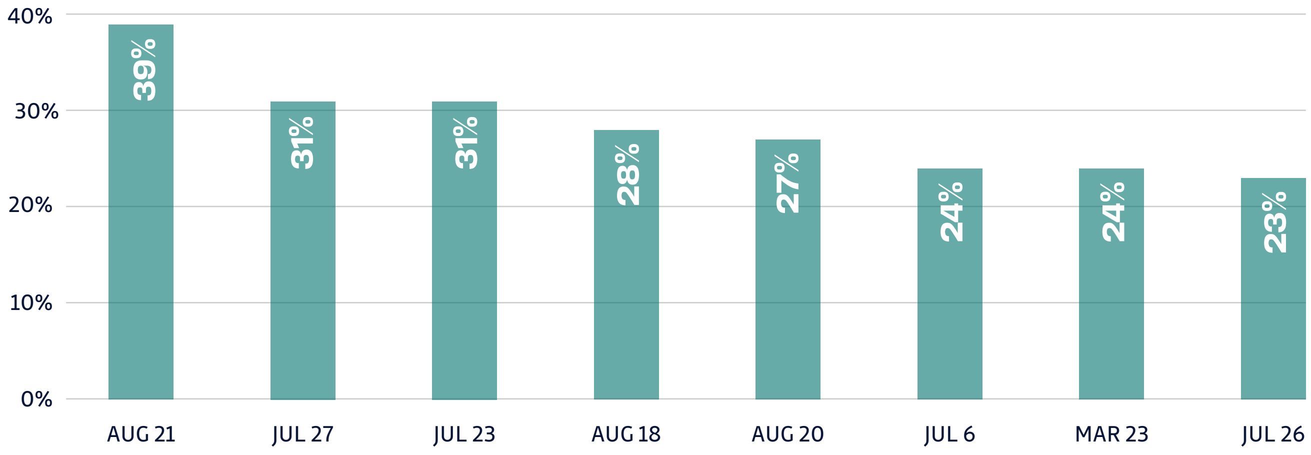
DIFFERENCE IN SECRETS/COMMIT FOUND ACROSS THE WEEK



The worst day of 2021 was
Saturday, November 20th
(+66% secrets/commit)

If we exclude weekends, most leaks happen
on holidays...

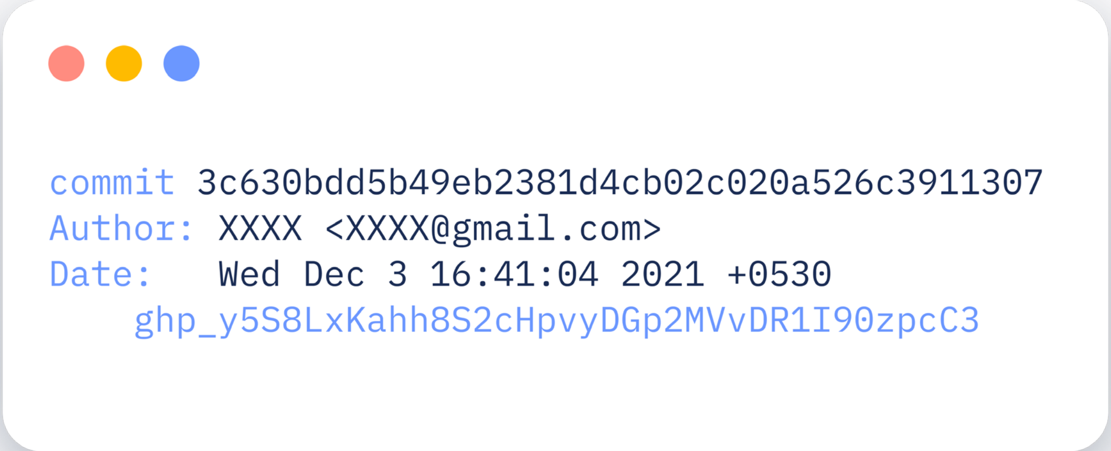
WORST DAYS OF THE YEAR, EXCLUDING WEEKENDS (Δ SECRETS/COMMIT)



We found more than

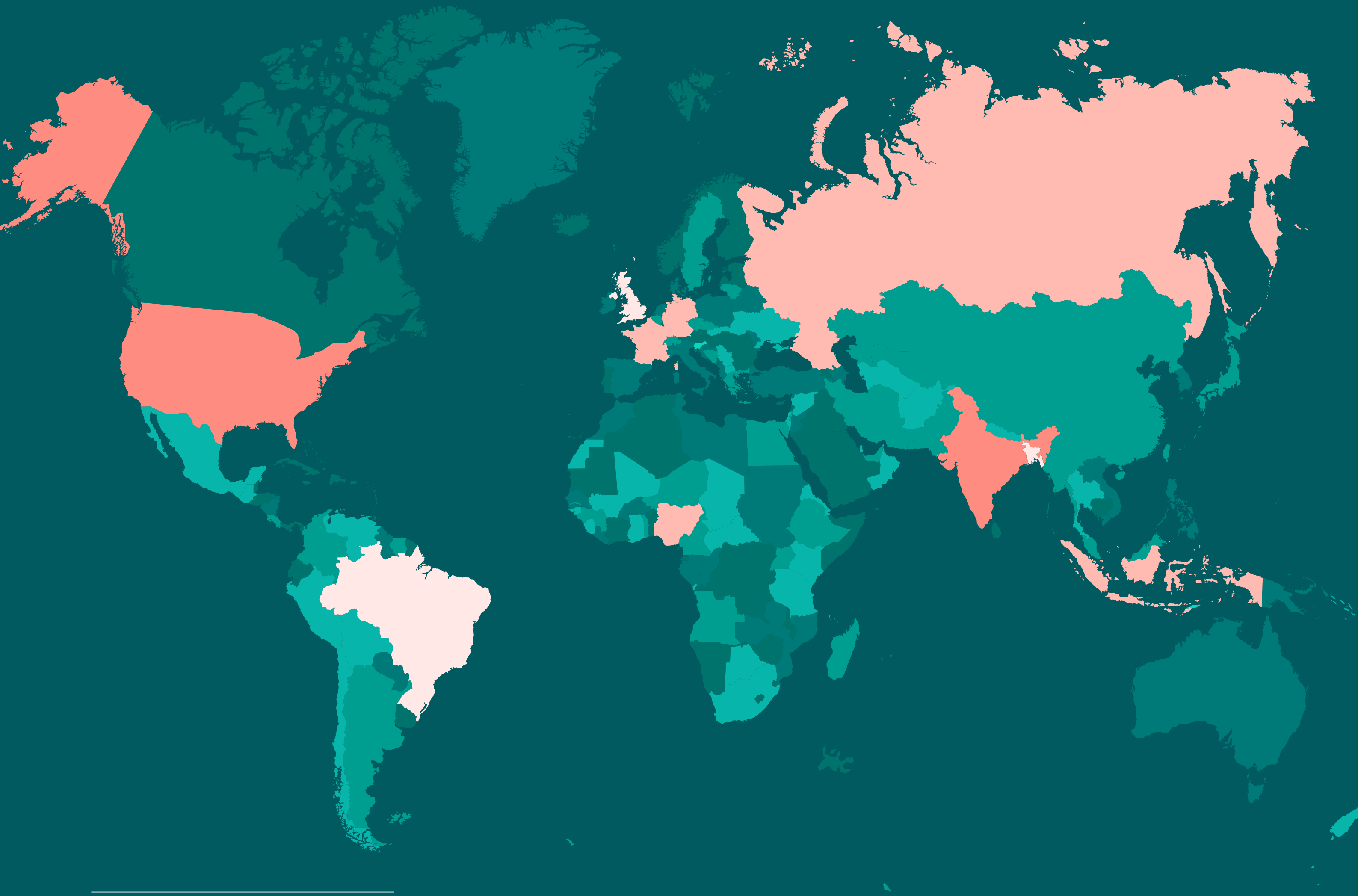
500 commit messages

containing GitHub
personal access tokens!



```
commit 3c630bdd5b49eb2381d4cb02c020a526c3911307
Author: XXXX <XXXX@gmail.com>
Date:   Wed Dec 3 16:41:04 2021 +0530
    ghp_y5S8LxKahh8S2cHpvyDGp2MVvDR1I90zpcC3
```

Where leaks come from



01	India
02	USA
03	Germany
04	France
05	Indonesia
06	Russia
07	Nigeria
08	Bangladesh
09	Brazil
10	UK

2021 breaches involving secrets leaks

Codecov

A favorite of many open-source projects, Codecov is a code coverage tool. Between January 31st and April 1st, it was compromised by attackers who were able to extract all of the environment variables of Codecov's customers. **Initial access was obtained by extracting a static GCP service account credential from one layer of Codecov Docker image**, which allowed them to tamper with a downstream CI script. Attackers were thus able to piggyback on Codecov to enter its users' private code repositories, exposing many more secrets. Read the full play-by-play.

Twitch

While most of the media attention has been focusing on streamers' revenues, GitGuardian conducted a deep-dive review to inspect the 6,000 git repositories of the Twitch codebase leaked this year. Despite lots of evidence demonstrating a certain level of AppSec maturity, **nearly 7,000 secrets were uncovered, including hundreds of AWS, Google, Stripe, and GitHub keys**. We triaged these secrets to explain how a malicious actor could easily leverage just a few of them to get a foothold into critical systems. To be fair, we found it quite disturbing that the main narrative about this leak was that it didn't present a security risk since no significant customer data was leaked. **It simply shows that the problem of secrets sprawl is largely underplayed**, even by some security experts.

Indian government

When white hat group Sakura Samurai started to scrutinize official Indian government endpoints under a vulnerability disclosure program, it didn't take them too long to find leaked credentials inside public-facing .git directories. In the end, **35 separate instances of exposed credential pairs were reported**, indicating a massive breach affecting government systems. As detailed in our play-by-play, the attack itself followed a well-proven methodology that is of relatively low sophistication, making it accessible to the widest range of hackers.

Internal Monitoring

In 2020, GitGuardian launched Internal Repositories Monitoring for Enterprise.

Monitoring thousands of repositories in real-time and scanning for the entire history of corporate codebases, we gained a realistic view of the state of application security in the DevOps era.

If there is a single conclusion to be drawn from the data, it is that

the amount of work required for both remediating real-time incidents and investigating leaks detected in the git history (which can still represent a threat) far exceeds current AppSec teams' capabilities.

> “Secrets detection is a very essential part of security. It’s one of the basics that you need to cover all the time. Otherwise, you’re going to expose your endpoints online and you’re going to suffer endless attacks. When it comes to application development, secrets detection is essential to a security program. You need to have it. Otherwise, you’ll fail.

Abbas Haidar, Head of InfoSec

Security teams are overwhelmed

On average, in 2021, **a typical company with 400 developers and 4 AppSec engineers** would discover

> 1,050 unique secrets leaked
> upon scanning its repositories and commits.

With each secret detected in 13 different places on average, the amount of work required for remediation far exceeds current AppSec teams capabilities (1 AppSec engineer for 100 developers)*.

* From TAG Cyber, see Methodology

1 AppSec engineer needs to handle

3,413

secrets occurrences

on average

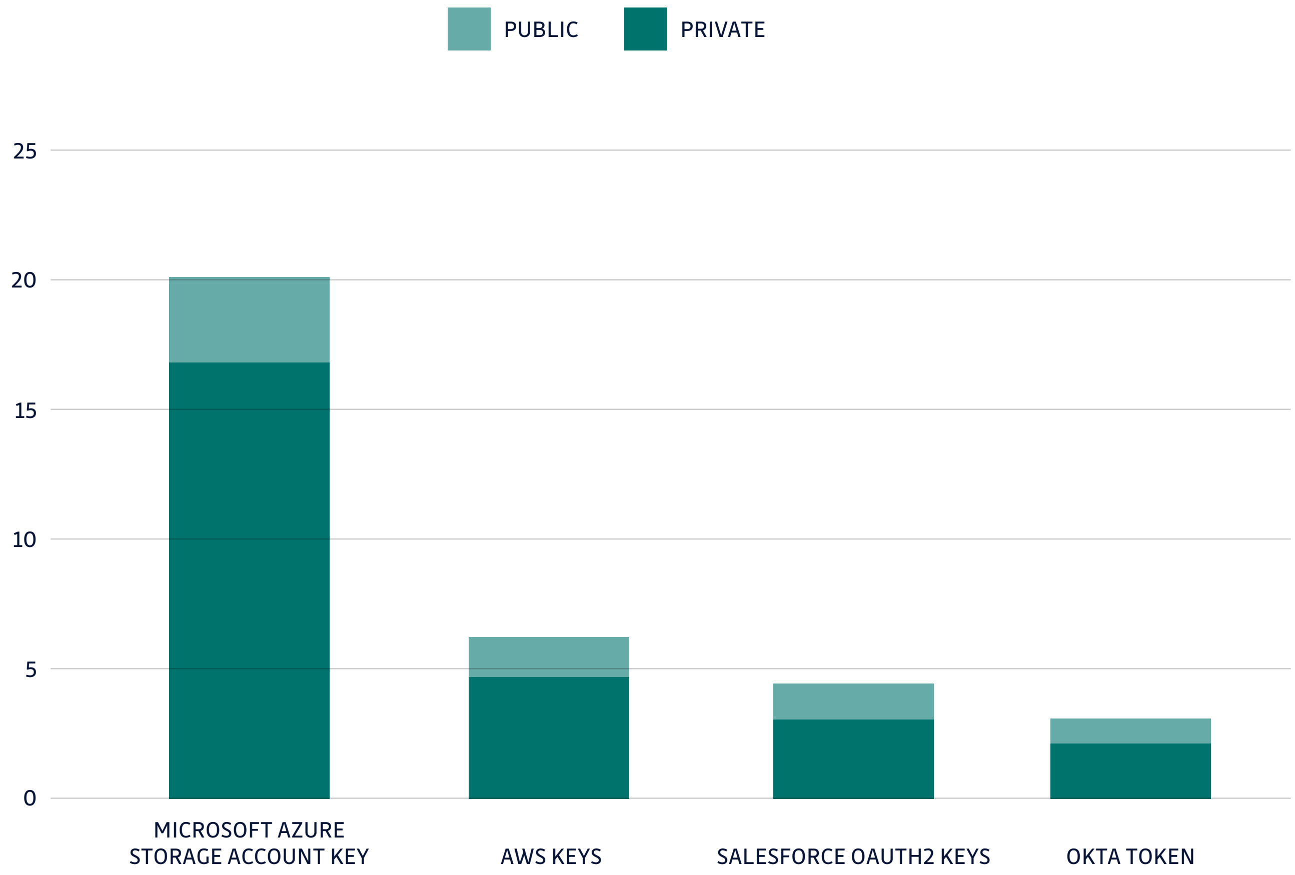
A false sense of secrecy

Our intuition that **private repositories permeate a false sense of secrecy**, causing even more leaks to occur compared to public ones, could be confirmed:

> On average, private repositories are 4x more likely to reveal at least one incident.

Not only private repositories are more likely to be affected, but they also reveal the real magnitude of secrets sprawl:

INCIDENTS DETECTED ON PUBLIC REPOSITORIES ARE ONLY THE TIP OF THE ICEBERG



AVERAGE NUMBER OF INCIDENTS PER ENTERPRISE REPOSITORY (SCANNED IN 2021) PER DETECTOR

> “I think people are getting more aware of secrets. [...]
> I think that it has had a positive impact on the culture itself. You’re only as good as the software you write, and you’re in for a world of hurt if you put the keys to the castle inside of that source code that could be somehow reverse-engineered. By separating the two, the source code and the keys, you’re one step ahead of that. I think it’s essential.

Blake, DevSecOps engineer

Recommendations

Private repositories fall too often victim of secrets sprawl, threatening security efforts and creating unnecessary remediation needs.

To prevent codebases from becoming hackers' playgrounds without overwhelming security teams, the focus needs to shift to **collaborative prevention**.

Secrets leaking in source code is unavoidable, but like other vulnerabilities, it is completely determined by endogenous factors: more developers, requiring access to more resources, building and deploying at a faster rate.

> It means that with enough **discipline and education**, coupled with **the right tools**, it is possible to drastically **improve the situation**, just like any technical debt.

Developer in the Loop

One core feature introduced in 2021, Developer in the Loop allows security engineers to share an incident with the developer who committed the secret. We are firmly convinced that a **Shared Responsibility Model** is key to enable application security at scale (security teams own the process, but developers are involved), and just a few months into the release the results already speak for themselves:

Involving the developer results in an incident closing rate

72% higher

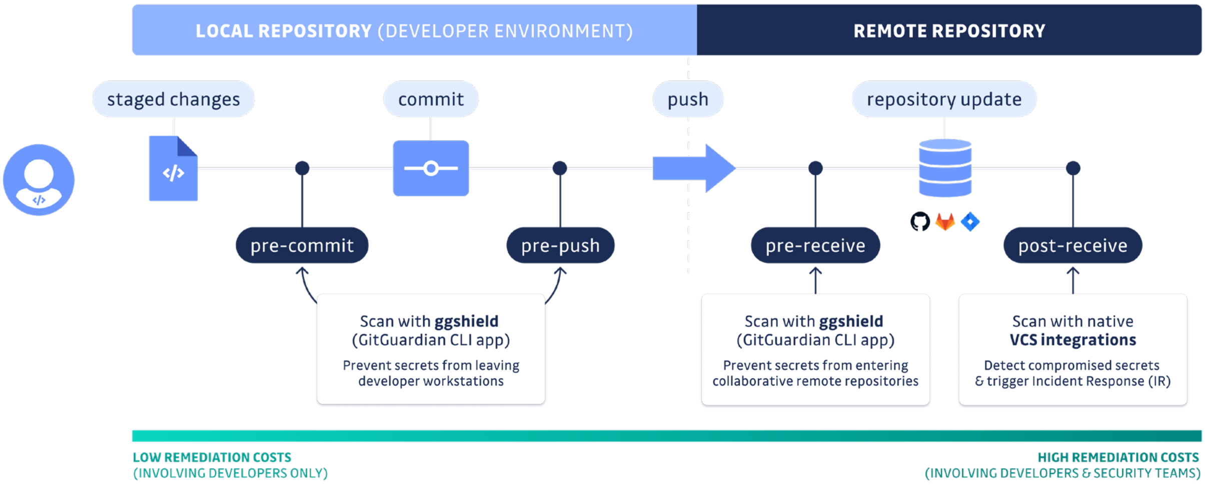
and a Median Time to Remediate

divided by 2

Solving the problem of secrets sprawl

Incidents detection and remediation can be shifted left at various levels to build a layered defense all across the development cycle. Here is a progressive approach to move forwards to a “zero secrets-in-code” policy:

1. Start by monitoring commits and merge/pull requests in real-time for all your repositories with native VCS or CI integration, where the ultimate threat lies (shift a team level).
2. Progressively enable pre-receive checks to harden central repositories against leaks, and “stop the bleeding”.
3. In the meantime, educate about using pre-commit scanning as a seatbelt (shift at developer level).
4. Plan a longer-term strategy to handle older incidents discovered by the git history scanning.
5. Implement a Secrets Security Champion program.


LOCAL REPOSITORY (DEVELOPER ENVIRONMENT)
REMOTE REPOSITORY
staged changes
commit
push
repository update
pre-commit
pre-push
pre-receive
post-receive
Scan with ggshield (GitGuardian CLI app)
Prevent secrets from leaving developer workstations
Scan with ggshield (GitGuardian CLI app)
Prevent secrets from entering collaborative remote repositories
Scan with native VCS integrations
Detect compromised secrets & trigger Incident Response (IR)
LOW REMEDIATION COSTS
(INVOLVING DEVELOPERS ONLY)
HIGH REMEDIATION COSTS
(INVOLVING DEVELOPERS & SECURITY TEAMS)

By integrating vulnerability scanning into the development workflow, security isn't a bottleneck anymore. You can help developers catch vulnerabilities at the earliest stage and considerably limit remediation costs. This is even more true for secrets detection, which is very sensitive to sprawling (as soon as a secret enters a version control system, it should be considered compromised and so requires remediation effort). On the other hand, you can reduce the number of secrets entering your VCS by better-educating developers while preserving their workflow.

Let's conclude

Secrets sprawl is a growing phenomenon, not only because more code is pushed, forked, and shared online every day, but also because the number of building blocks making up an application is increasing (cloud infrastructure, managed databases, SaaS applications, open-source components, internal microservices…).

As the sprawling accelerates, version control systems are quickly becoming a top target for hackers looking to start a supply chain attack, as seen in multiple breaches last year. Compromising hardcoded secrets requires no special skills, and the proliferation of leaked secrets in public GitHub (which more than doubled since 2020) **is a red flag for many application security professionals**. Public repositories must be therefore included in the safeguarded corporate perimeter.

The situation is also true on the internal side. Private repositories hide a huge amount of (often forgotten) secrets that could one day be used for fraudulent purposes. Unfortunately, even if they are aware of it, AppSec are overwhelmed by the amount of work to be done, either to remediate incidents on-the-fly or to dig through the stack of older but still present ones.

There is an urgency to remove secrets from source code, but to do so requires adopting the right mindset: our experience has shown that the only reasonable approach to deliver secure software at scale is to share the application security responsibility between developers, security, and ops. Enabling this model is our mission.

If you want to learn more about how GitGuardian solutions can help you improve on code security, don't hesitate to **contact us**.

About GitGuardian

The new ways of building software create the necessity to support new vulnerabilities and new remediation workflows. These needs have emerged so abruptly that they have given rise to a young and highly fragmented DevSecOps tooling market. Solutions are specialized based on the type of vulnerabilities being addressed: SAST, DAST, IAST, RASP, SCA, Secrets Detection, Container Security, and Infrastructure as Code Security. However, the market is fragmented and tools are not well-integrated into the developers' workflow.

GitGuardian, founded in 2017 by Jérémy Thomas and Eric Fourrier, has emerged as the leader in secrets detection and is now focused on providing a holistic code security platform while enabling the Shared Responsibility Model of AppSec. The company has raised a $56M total investment to date.

With more than 150K installs, GitGuardian is

THE #1 SECURITY APPLICATION ON THE GITHUB MARKETPLACE.

Its enterprise-grade features truly enable AppSec and Development teams in a collaborative manner to deliver a secret-free code. Its detection engine is based on 350 detectors able to catch secrets in both public and private repositories and containers at every step of the CI/CD pipeline.

Methodology

Secrets detection engine

Taking into account statistical outliers, GitGuardian detected over 6M secrets on public GitHub in 2021. To accurately estimate the global growth of this figure compared to the previous year, we revised upwards our 2020 estimate (2M secrets detected) which was conservative. The reference we took for the global volume of secrets found in 2020 is 2.66M, leading to an increase of +125% YoY.

GitGuardian's secrets detection engine has been running in production since 2017, analyzing billions of commits coming from GitHub. From day one we began to train and benchmark our algorithms against the open-source code. It allowed GitGuardian to build a language-agnostic secrets detection engine, integrating new secrets or new ways of declaring secrets really fast while keeping a really low number of false positives. We have developed the vastest library of specific detectors being able to detect more than 350 different types of secrets. Learn more about the inner workings and performance benchmarking of our detection engine in our blog.

AppSec to Developers ratio

The TAG Cyber Analyst Team conducted a study that resulted in an average AppSec to developers ratio of 1:200, with a lower bound of 1:100. For the 2022 State of Secrets Sprawl, we used this last figure in a conservative approach.

From the TAG Cyber report: *"The TAG Cyber analysts collected data from present or former CISOs in telecom, insurance, finance, mobility services, R&D, and technology. In each case, the question was posed regarding rough estimates between developers and application security experts. Answers came back as high as 1000 to 1 and as low as 100 to 1. The average came to a roughly 200 to 1 ratio."*

Docker Hub

For this study, we scanned:

- **10K** random Docker images pulled from Docker Hub
- **65,104 tar files**: 65,103 layers + images' metadata
- **143 million** documents.

Internal Monitoring

When users choose to monitor their repositories with GitGuardian for the first time, they are offered to operate a historical scan over all their repositories (the detection engine is the same as used for GitHub Public Monitoring).

In 2021, GitGuardian Internal Monitoring detected on average 13,635 secrets occurrences per Enterprise account.

For the 60 most common secrets (see Appendix), the percentage of private repositories exposing at least one incident is 10.6%, and the percentage of public repositories exposing at least one incident is 2.8%. Private repositories were 3.8 times more likely to expose at least one secret than public repositories.

The average number of incidents per enterprise repository (private and public) and per detector was computed on repositories scanned at least once in 2021. For a vast majority of our detectors, incidents were only detected on private repositories. On the graph, we chose to display only the top four detectors with a non-null average for public incidents.

Appendix

List of common secrets:

- Alibaba Cloud Keys
- Artifactory Token
- Auth0 Keys
- AWS Keys
- Bitbucket Keys
- Cloudflare API Credentials
- Confluent Keys
- Datadog API Credentials
- DB2 Credentials
- DigitalOcean OAuth Application Keys
- DigitalOcean Spaces Keys
- DigitalOcean Token
- Dropbox App Credentials
- Facebook Access Token
- Facebook App Keys
- GitHub Access Token
- GitHub App Keys
- GitHub OAuth App Keys
- GitLab Enterprise Token
- GitLab Token
- Google Cloud Keys
- HubSpot API Key
- Intercom Access Token
- Jira Basic Auth
- Kubernetes Cluster Credentials
- LDAP Credentials
- MariaDB Credentials
- Microsoft Azure Storage Account Key
- Microsoft Teams webhook
- MongoDB Credentials
- MSSQL Credentials
- MySQL Credentials
- npm Token
- NuGet API Key
- ODBC Connection String
- Okta Keys
- Okta Token
- Oracle Credentials
- PayPal OAuth2 Keys
- PostgreSQL Credentials
- Python Package Index Key
- Redis Credentials
- Salesforce Oauth2 Keys
- Salesforce Refresh Tokens
- SendGrid Key
- Slack App Token
- Slack Application Credentials
- Slack Bot Token
- Slack Signing Secret
- Slack User Token
- Slack Webhook URL
- SMTP credentials
- Snowflake Credentials
- Splunk Authentication Token
- Tencent Cloud Keys
- Terraform Cloud Token
- Twilio Keys
- Twilio Master Credentials
- VISA Basic Auth
- Zendesk Token

GitGuardian

www.gitguardian.com