

[SOLUTION BRIEF]

Leaked keys, found before attackers do.

Keys, tokens and credentials found across your repositories and their full history, correlated with the code, cloud and application findings around them.

[THE PROBLEM]

A credential committed once lives in git history forever, long after the line is deleted. Exposed secrets and configuration account for roughly one in six findings from Parameter's autonomous pentests, and a processor key or cloud access key in a repository is the shortest path from source access to production data. Regex scanners find strings; teams need to know which of them matter.

WHAT SECURITY TEAMS NEED

- 01 **History, not just HEAD**
Every branch and every commit, so a key deleted last year is still found.
- 02 **Context on impact**
Which cloud resource, data store or third-party service the credential unlocks.
- 03 **Caught at the PR**
New secrets flagged inline before they merge, not after they land in history.

[BY THE NUMBERS]

17%

Of pentest findings are exposed secrets or config

PR + history

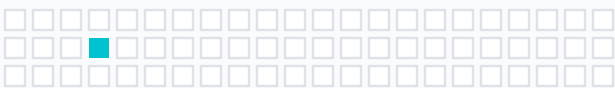
From the first commit to the next pull request

1 queue

Secrets alongside code, cloud and app findings

Old way

A regex sweep of HEAD before the audit.



Parameter

Every commit, every branch, every pull request.



BUILT BY THE TEAM THAT SECURED

Apple · Google · Microsoft · U.S. Department of Defense · T-Mobile · Henkel · Epic Games · AT&T · Yahoo

BACKED BY Y COMBINATOR

[HOW IT WORKS]

Scan history. Catch at the PR. Correlate the blast radius.

01

Connect

Connect GitHub, GitLab or Bitbucket Cloud with read-only access. Parameter scans the repositories you select, including every branch and the full commit history.

02

Detect

Keys, tokens and credentials are identified across repositories and history. Sentinel flags new secrets inline on every pull request before they merge.

03

Correlate

Findings are placed in the shared platform queue and correlated with the cloud resources and data stores the credential reaches, so remediation starts with the highest-impact key.

[SOLUTION ARCHITECTURE]

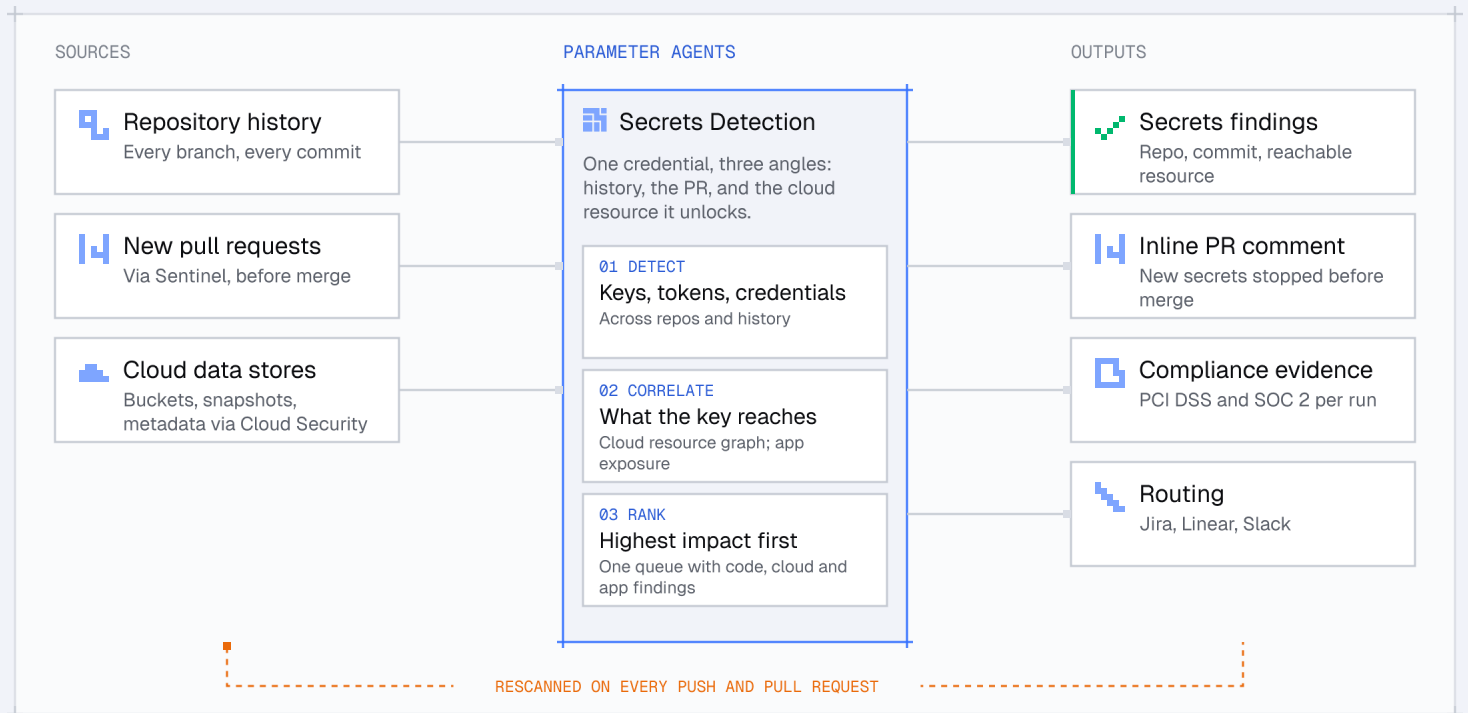


FIG. 01 Secrets Detection flow. Repository history and new pull requests are scanned for credentials; results are correlated with Cloud Security's resource graph and the Pentesting Agent's exposed-configuration findings in one queue.

Across repos and history

Leaked keys, tokens and credentials found across your repositories and their history, not only the current tree.

Inline before merge

Sentinel treats leaked secrets as a first-class vulnerability class and comments on the pull request that introduces one.

Impact through correlation

Cloud Security maps secrets and leaked keys in data stores and snapshots; the Pentesting Agent surfaces exposed configuration in the running app. All of it lands in one ranked queue.

[KEY CAPABILITIES]

Three angles on one credential.



Repository history scanning

Every branch and commit in connected repositories, so deleted credentials are still surfaced.



Pull-request detection

New keys, tokens and credentials flagged inline by Sentinel before they merge.



Cloud data-store exposure

Secrets and leaked keys in public buckets, snapshots and resource metadata, from the Cloud Security module.



Exposed configuration in the app

Configuration and credentials reachable in the running application, proven by the Pentesting Agent.



Routing to owners

Findings routed to Jira, Linear or Slack and tracked in the shared platform queue.



Compliance evidence

Processor keys in git history called out for PCI DSS and SOC 2 evidence per run.

[PRODUCT]

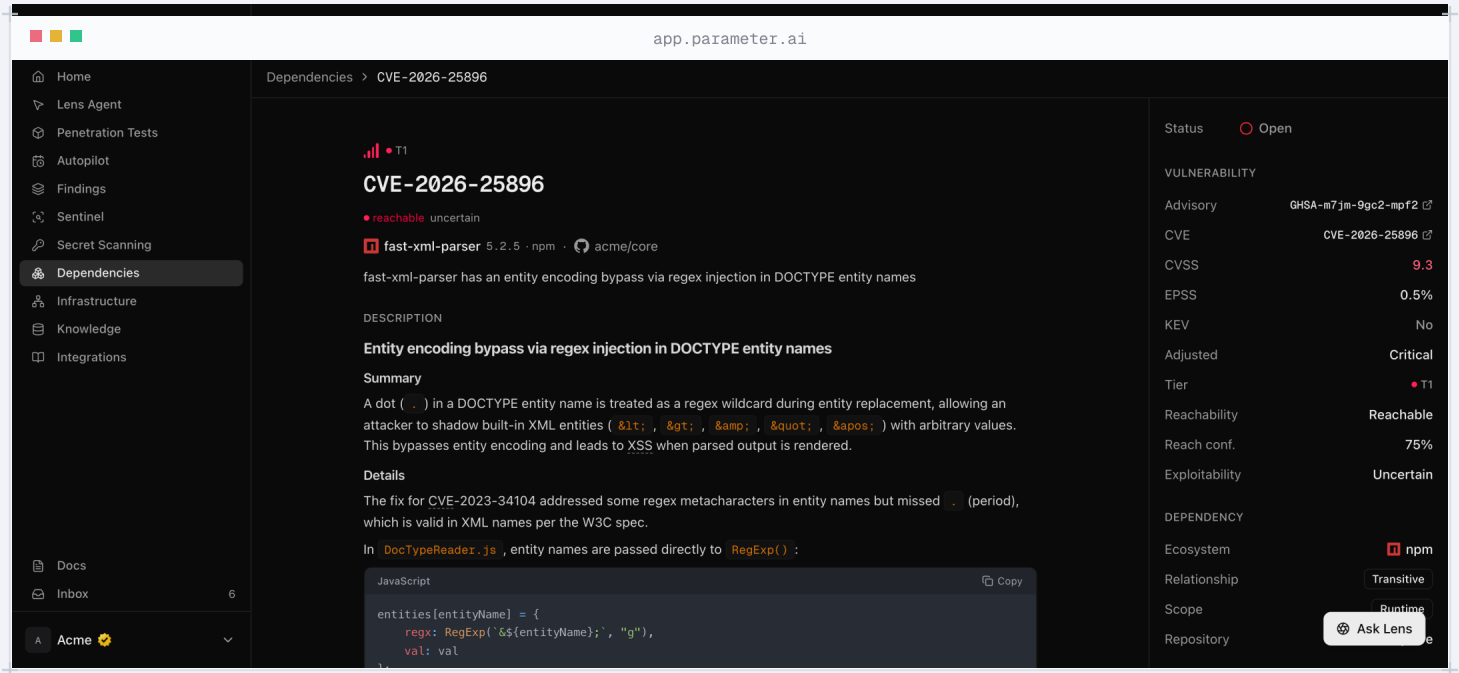


FIG. 02 Secrets findings in the platform queue, with the repository, commit and the cloud resource each credential reaches.

[TECHNICAL SPECIFICATIONS]

SOURCE CONTROL	GitHub, GitLab, Bitbucket Cloud
SCOPE	Selected repositories, all branches, full commit history
ACCESS	Read-only repository access; no agent to install
PR COVERAGE	Via Sentinel, inline on every new pull request
CORRELATION	Cloud Security resource graph; Pentesting Agent findings
ROUTING	Jira, Linear, Slack; shared platform queue
EVIDENCE	Repository, commit and reachable resource recorded per finding
COMPLIANCE	PCI DSS and SOC 2 evidence per run

WHERE SECRETS ARE COVERED

- **Secrets Detection**
Repositories and full git history, on every connected repo.
- **Sentinel**
Leaked secrets as a vulnerability class on every pull request.
- **Cloud Security**
Secrets and leaked keys in data stores, snapshots and resource metadata.
- **Pentesting Agent**
Exposed secrets and configuration proven in the running application.

[INCLUDED WITH]

Single App

Repository connection with source access (whitebox)

Rightsized

Scope set automatically from repos; multi-repo

Continuous

Every push and every pull request

[WHY PARAMETER]

- ✓ **History included**

A credential deleted from HEAD is still a credential. Parameter scans the full history.
- ✓ **Impact, not just presence**

Correlation with cloud resources tells you what a key unlocks, not just that it exists.
- ✓ **Stopped at the PR**

Sentinel flags new secrets inline before they ever reach history.
- ✓ **Part of one platform**

Secrets, code, cloud and application findings ranked together in one queue.

[PART OF THE PARAMETER PLATFORM]

Secrets Detection is delivered through the same repository connection as Sentinel and Supply Chain. Connect once and every module runs against the same code.

See first findings in 24 hours.

ABOUT PARAMETER

Parameter is an AI-powered security platform that continuously finds and validates vulnerabilities before attackers can exploit them. Autonomous agents pentest like a real adversary across your code, cloud and dependencies, with every finding proven. Formerly Hex Security, Parameter is backed by Y Combinator and built by a team that has secured Apple, Google, Microsoft, the U.S. Department of Defense, T-Mobile, Henkel, Epic Games, AT&T and Yahoo.

TRUST AND COMPLIANCE

SOC 2 Type I (Type II observation underway)
HIPAA BAA available
GDPR DPA with SCCs
Data processed in the US
parameter.ai · support@parameter.ai · app.parameter.ai