Gitleaks Secret scanner

CONCEPTS

> Exit code 1 means leaks were found, 0 means clean. That is what makes it usable as a CI or hook gate.

Source git history, directory, stdin
↓ match
Rules default or .gitleaks.toml
↓ filter
Allowlists paths, regexes, baseline, allow comments
↓ report
Findings log, json, sarif, csv, junit

> detect and protect are deprecated since v8.19.0 and hidden from --help. Use git, dir and stdin instead.

SCAN MODES

`gitleaks git`
Scan commit history of the current repo

`gitleaks git` *`<repo-path>`*
Scan another repository

`gitleaks dir` *`<path>`*
Scan a folder or single file

`cat` *`<file>`* **`| gitleaks stdin`**
Scan piped data

`gitleaks git --log-opts="`*`<opts>`***`"`**
Pass git log options to limit commits

`gitleaks git --log-opts="--all"`
Include all branches and tags

`-v / --verbose`
Print each finding in detail

REPORTS

`-r` *`<file>`*
Write findings to a report file

`-f json / csv / junit / sarif`
Choose the report format

`--redact`
Mask secrets in logs and stdout

REPORTS (cont.)

`--exit-code` *`<n>`*
Exit code to use when leaks are found

BASELINE AND IGNORES

`gitleaks git -r baseline.json`
Create a baseline from today's findings

`-b` *`<baseline.json>`*
Report only findings not in the baseline

`// gitleaks:allow`
Comment on a line to skip its findings

`.gitleaksignore`
Repo file listing finding fingerprints to skip

CONFIGURATION

`-c` *`<file>`*
Use this config file

`.gitleaks.toml`
Config picked up from the scan target

`[extend] useDefault = true`
Keep default rules and add your own

`[[rules]] id, regex, keywords`
Define a custom detection rule

ALLOWLISTS

`[[allowlists]]`
Global allowlist block

`paths = ['''`*`<regex>`***`''']`**
Skip files by path regex

`regexes = ['''`*`<regex>`***`''']`**
Skip findings matching a regex

HOOKS AND CI

`repo: https://github.com/gitleaks/gitleaks`
pre-commit repo entry; pin rev, id: gitleaks

`gitleaks/gitleaks-action`
GitHub Action that runs the scan