Secrets Scanning Checklist

6 Key Criteria for Developer-First Secrets Scanning Solutions

Hardcoding secrets enables developers to seamlessly access or authenticate the services needed to build and deploy applications. But those secrets, if not stored securely, present a huge risk. Hardcoded secrets—such as passwords, credentials, API keys, and important tokens—can be exposed in source code, build logs, infrastructure as code (IaC), repositories, and more. If important credentials fall into the hands of a bad actor, they could be used to gain privileged access to leak data, tamper with code, steal sensitive information, shut down services, or run up exorbitant fees.

Because hardcoded secrets are so common, particularly in matrixed development organizations and within cloud-native applications, they're becoming a popular target for bad actors.

Secrets security solutions can help you get ahead of these risks and adopt a well-rounded code security program. But not all secrets scanning tools provide the breadth or depth of coverage you need to identify and protect your secrets.

In this checklist, we'll cover the six key criteria you should look out for when evaluating a secrets scanning solution.

1. Scans Both Application Code and Infrastructure as Code Files

Secrets can appear in any type of code in your organization, so your secrets scanning solution should be able to scan all of the major file types, such as container images, IaC templates, and source code. Without complete scanning, you'll lack visibility into where exposed secrets exist within your organization.

Let's say that your secrets scanning solution scans for secrets in IaC but not in application code. Your developers will get visibility into exposed secrets in their IaC templates and files so they can remediate those issues, but there is still the risk of exploit if you have application code elsewhere in your codebase.

But when your secrets scanner scans both application code and IaC for secrets, you'll get comprehensive coverage and visibility into all potential areas where secrets could be hiding. Plus, your secrets tooling is slightly less noisy because it accounts for the context of an IaC file around the secret.

Recommended secrets scanning solution criteria:

- Scans for secrets in both IaC and application code
- Augments secrets scanning with the context of an IaC file around the secret

2. Developer-Friendly Integrations

Involving developers in secrets security is the most efficient way to identify, remediate, and prevent exposed credentials. To ensure that you're engaging developers in this process as seamlessly as possible, your secrets scanning solution must natively integrate with developer tools. This way, you can empower your developers to actually prevent secrets security issues, all while reducing the context-switching that can kill productivity and developer well-being.

When evaluating secrets scanning solutions, make sure to look for a solution that surfaces feedback directly into developer tools and DevOps workflows.

Recommended secrets scanning solution criteria:

- Natively integrates into existing developer tools, such as VCS and IDEs
- Integrates with DevOps workflows such as CI/CD pipelines
- Surfaces both the exposed secret and the context around that secret to streamline risk prioritization and remediation
- Blocks secrets from being pushed to a repository before a pull request is opened via a pre-commit hook and surfaces exposed secrets as part of a pull request scan

3. A Multidimensional Approach to Secrets Scanning

Secrets using regular expressions—such as access tokens, API keys, encryption keys, OAuth tokens, certificates, and others—are the most commonly identified exposed credentials. With regular expression scanning, secrets security solutions identify if a given string follows the pattern of other secrets of that type, like an AWS Access key. But secrets can also take a number of other, less-predictable forms that are harder to identify systematically.

In addition to regular expression scanning, a complete solution should include keyword scanning, wherein strings commonly associated with a secret, such as "password," are picked up. A more advanced form of secrets security involves scanning for high-entropy patterns, which evaluates potentially exposed credentials to determine if they differ enough from real language to potentially be a secret, such as `EuN21!HHvaS%JPTQU&cx`. This is the most challenging form of scanning as it can result in noisy false positives, which we'll cover in more detail in the next section.

The only way to identify all of your exposed secrets is to employ a solution that includes regular expression, keyword, and entropy-based scanning. However, many secrets scanning tools scan disparately for only one type of secret or at only one phase of the development lifecycle, which can create noise from false positives and can miss secrets altogether.

A secrets security solution's efficacy also depends on both the breadth of the secret detectors it leverages and the depth of its scanning. If your solution leverages a large signature library to detect and alert on the wide array of secrets with known, predictable expressions, then you can be sure you're getting the breadth of scanning you need.

Recommended secrets scanning solution criteria:

- Leverages regular expression, keyword, and entropy-based scanning
- Employs domain-specific secret detectors
- Built on top of a large signature-based policy library
- Continuously scans for exposed credentials across the development lifecycle, from build-time to runtime
- Scans all source code files and version histories to identify secrets buried deep within your codebase

4. Fine-Tuned High-Entropy Pattern Recognition

While scanning for regular expressions and keywords can identify a significant percentage of exposed credentials, these scanning methods will miss secrets that don't have consistent or identifiable patterns. If you only employ these signature-based scanning methods, you'll miss secrets like random string usernames and passwords. To avoid this problem and get multidimensional secrets scanning, it's important that your secrets scanner includes entropy-based secrets scanning that's based on a fine-tuned entropy model.

When an entropy-based secrets scanning tool analyzes a string, it determines how different that string is from any word in the English language. If that string is very different from real English words, the tool will flag that string as an exposed secret. While this type of scanning can flag secrets that other forms of scanning miss, entropy-based scanning may also introduce some false positives. Let's say you use a very long name for a variable, such as `redirectForOptimization`. Because that variable is clearly not a word, basic entropy-based scanners would probably incorrectly flag that variable as a secret.

There are a few ways to reduce this false positive rate. To start, you'll want to choose a solution that fine-tunes the entropy model by leveraging the string's context to more accurately identify complex secret types. You should also look for a secrets security solution that combines keyword and entropy-based scanning. Solutions with a multidimensional approach have the benefit of lowering your false positive rate while also providing more complete coverage to ensure you're finding all your exposed secrets.

Recommended secrets scanning solution criteria:

- Includes both signature-based and entropy-based secrets scanning
- Entropy model accounts for the string's context to identify complex secrets types and reduce the false positive rate

5. Part of a Complete Code Security Solution

Secrets scanning is just one component of a complete code security solution, and given the complexity of adopting separate security tooling for each component of a cloud-native codebase, it's most efficient to consolidate your code security by adopting a single code security solution.

Adopting multiple code security use cases—such as secrets scanning, IaC security, software composition analysis (SCA), and others—introduces sprawl, leaves coverage gaps, and makes it challenging to address the fact that secrets, vulnerabilities, and misconfigurations can appear across the entire development lifecycle. Surfacing security findings across disparate tools also results in noise, further distracting developers from their work, and when teams utilize disparate security tools that address infrastructure and application security separately, it's challenging to prioritize issues based on exposure.

Recommended secrets scanning and code security solution criteria:

- Constitutes one component of a singular code security solution, which consolidates tool sprawl and minimizes coverage gaps
- Continuously scans for exposed secrets, misconfigurations in IaC files, and vulnerabilities in open source code and container images
- Gives complete visibility into all security issues and dependencies via a Supply Chain Graph
- Monitors and prevents misconfigurations and vulnerabilities throughout the development lifecycle to give you code-to-cloud coverage

6. Runtime Connection with Cloud Infrastructure Entitlement Management Solutions

To seamlessly remediate over-privileged identities and secrets sprawl in the cloud, your organization needs a secrets scanning solution that's aligned with a Cloud Infrastructure Entitlement Management (CIEM) solution. CIEM solutions enable organizations to manage identities and privileges in cloud environments. With CIEM, you can understand which access entitlements exist across cloud and multicloud environments. Once you gain that visibility, you can then identify and mitigate risks resulting from entitlements that are both unused and grant a higher level of access than they should. A comprehensive CIEM solution includes functionality such as net-effective permissions calculation to give you visibility into who can do what within your organization. Complete CIEM solutions also reduce permission sprawl, leverage advanced ML and behavior-based anomaly detection policies, and combine runtime and IaC data.

Having one solution for both secrets security and CIEM empowers your organization with comprehensive cloud identity security. You'll get visibility into who can access sensitive resources like key vaults and secrets, and you can understand how secrets are being used. Plus, you'll reduce the number of unused and other risky permissions, which can lead to security issues down the line.

Recommended secrets scanning and CIEM solution criteria:

- Aligns secrets scanning feedback with CIEM functionality such as visibility, tracking, and right-sizing
- Gives context into which secrets are being used by which identities
- Reduces unused and risky permissions

Keeping these criteria in mind will help ensure that you're getting the full-stack, multidimensional secrets scanning you need to prevent the risks associated with exposed credentials.

3000 Tannery Way
Santa Clara, CA 95054

Main:	+1.408.753.4000
Sales:	+1.866.320.4788
Support:	+1.866.898.9087

www.paloaltonetworks.com