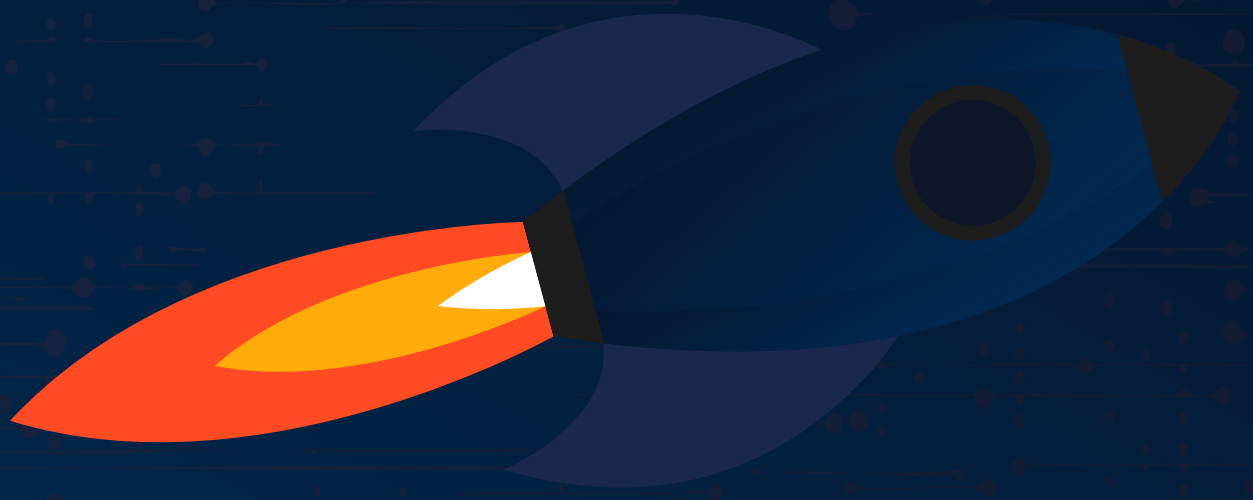
SCANOSS



SCA for the AI Development Lifecycle

running in production at enterprise scale

KEY TAKEAWAYS

Compliance checks run at generation

Compliance checks run at the moment Claude Code writes AI-generated code, not after commit.

Live across 16,000 developers

Live across 16,000 developers at a global enterprise, no pilot required.

Non-compliant code is flagged

Non-compliant open source is blocked or flagged for legal review automatically.

THE SITUATION

AI-generated code moved faster than traditional SCA could follow

AI coding assistants now write a large and growing share of enterprise code, and that code routinely contains open source fragments — copied, adapted, or model-generated — carrying licence obligations and security risks that never appear in a dependency manifest. Traditional SCA tools run in CI or after commit, hours or days after the code is written. By the time a scan runs, the obligation is already sitting in the codebase, and remediation becomes a retrospective legal exercise rather than a development-time decision.

THE SOLUTION

Moving the compliance check to the moment of generation

SCANOSS built a **compliance gate that runs inside Claude Code** itself, matching every line the assistant writes against the SCANOSS knowledge base using true snippet-level detection, a capability dependency-only scanners cannot perform.

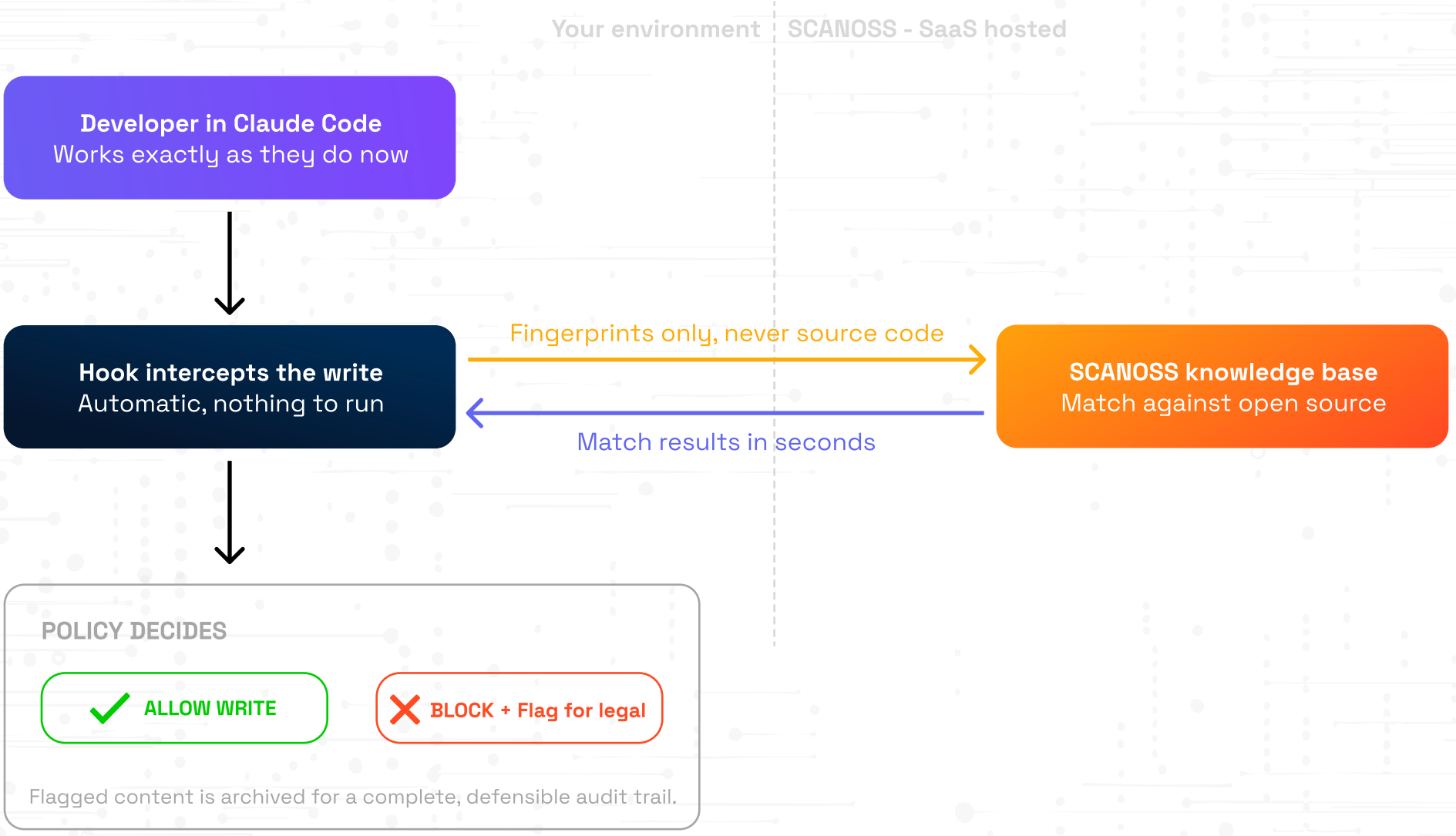
The workflow runs in four steps. A developer works in Claude Code exactly as they do now. A lightweight hook intercepts the code the moment it is written to a file, with no manual step and nothing for the developer to remember or run. The candidate code is fingerprinted and sent to the SCANOSS API; only fingerprints are matched, never raw source, so intellectual property never leaves the environment as plaintext. SCANOSS matches the fingerprint against its Knowledge Base and returns a result in seconds, including whether a match exists, how many lines matched, and the confidence of that match.

A configurable policy then decides the outcome: allow the write, or block it and flag the snippet for legal review, with optional archiving of flagged content for a complete, defensible audit trail. This is a fully automated compliance gate operating silently inside Claude Code's own workflow, catching non-compliant open source at the point of creation rather than discovering it weeks later in an audit.

Because the architecture is SaaS-hosted, there is no knowledge base to build and no infrastructure to stand up. The deployment runs on SCANOSS's existing hosted KB and scanning service, with capacity that flexes elastically against real usage patterns across time zones and workdays.

> **16,000 developers, and compliance checks now run at the exact moment Claude Code writes a line of code — not days later in a scan.**

SCA for the AI Development Lifecycle



THE OUTCOME

Live in production across 16,000 developers

The company deployed SCANOSS across 16,000 developers, running real-time scans on every piece of code Claude Code writes, inside the developer's own flow. There was no pilot, no custom data preparation, and no new platform to adopt: the deployment runs on SCANOSS's existing knowledge base and existing tooling, and reached production in about a week — built for what SCANOSS terms the **AI Development Lifecycle**, in which design-to-merge compresses from weeks into minutes. Controls designed for the human software development lifecycle — periodic audits, post-commit scans, manual review queues — cannot keep pace with code generated and committed at machine speed, which is why the company needed a compliance layer engineered to operate at the point and pace of AI generation itself, rather than retrofitted onto it. Enterprise tooling around this capability extends to **SBOMs and AIBOMs in CycloneDX and SPDX, cryptographic-algorithm detection, AI-artefact detection, and audit logging across IDE, CI, webhook, and SDK integrations.**

Ready to enhance visibility, mitigate risks, and future-proof your operations?

sales@scanoss.com