Scribe

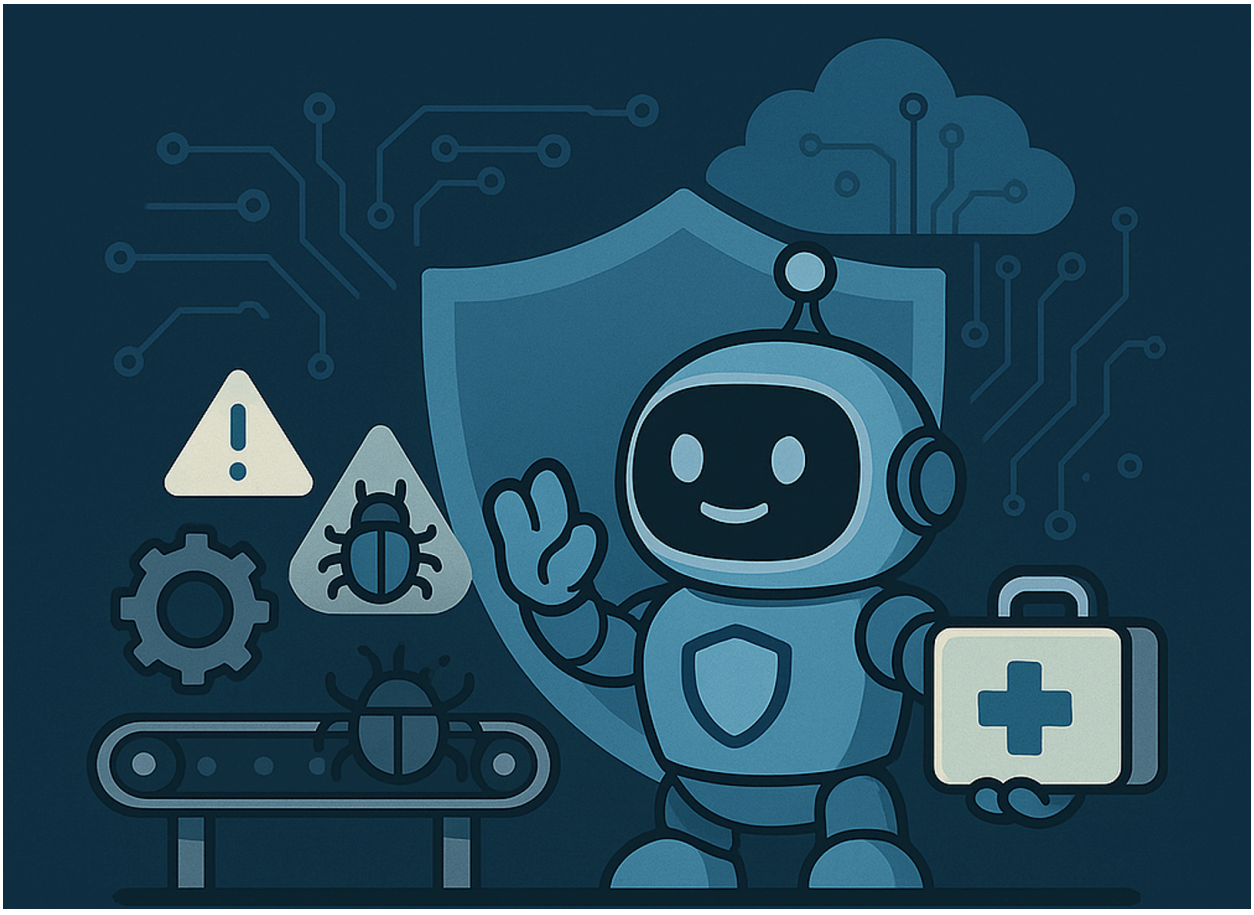
From Continuous Assurance to Agentic AppSec

How Scribe Security Protects Your AI-Driven Software Factory

Additional information is available at https://scribesecurity.com/

From Continuous Assurance to Agentic AppSec - Software Security for The AI Era

How Scribe Protects Your AI-Driven Software Factory



Abstract

Software supply chain risk has outpaced traditional application security. CI/CD pipelines, third-party components, and AI-generated code now change faster than AppSec teams can inspect, review, and remediate. Scribe Security addresses this gap with continuous, evidence-based assurance across the software development lifecycle (SDLC). The platform automates the collection, signing, and verification of SDLC evidence of all types; builds a knowledge graph that links artifacts, identities, and actions; enforces policy-as-code guardrails; and now, in the AI era, orchestrates a network of intelligent agents to analyze risk, prioritize fixes, and remediate at scale. This paper explains the market problem, Scribe's data and

architecture advantage, its evolution to an AI-centric model, and practical outcomes for CISOs, product security leaders, and DevSecOps practitioners.

---

1) Market Reality: Risk Scales Faster Than Security

Modern development velocity breaks old security models. Every day, organizations ship code from dozens of repositories through multiple pipelines, into registries and clusters across clouds. The attack surface now includes source control, build scripts, container bases, deployment manifests, signing keys, and the people and automations touching them.

Three shifts make the gap unmanageable by hand:

1. **Exploding component reuse:** Open source and third-party services dominate modern apps; one dependency can expose thousands of builds.

2. **Automated pipelines as targets:** CI/CD systems are now prime intrusion points; tampering here compromises everything downstream.

3. **AI-generated code at scale:** LLMs accelerate output, but also accelerate the volume of insecure code, misconfigurations, and leaked secrets.

Compliance adds pressure. Governments and enterprises now expect verifiable proof: SBOMs, provenance, signed attestations, and process controls aligned to frameworks like SLSA, NIST SSDF, DORA, FedRAMP, CRA, FDA, PCI. Point-in-time scans and spreadsheets cannot satisfy these demands reliably.

Conclusion: The industry must move from “detect and patch later” to **continuous assurance**: capture evidence as software is created, **prove** integrity and policy conformance before promotion, and remediate quickly with minimal human toil.

2) Scribe's Core Approach: Evidence, Integrity, Policy, and Speed

Scribe's platform (ScribeHub) operationalizes continuous assurance across the SDLC:

- **Automated evidence collection**
  Scribe collectors integrate with SCMs, build systems, registries, and clusters to gather multi-stage SBOMs, scanner results, pipeline posture, and process context. Only evidence - **not source code** - is collected.

- **Signing, attestations, and provenance**
  Evidence is signed (enterprise PKI or Sigstore), transformed into machine-readable attestations (including in-toto/SLSA provenance), and validated at each stage: commit, build, test, package, deploy.

- **Tamper-proof integrity graph**
  Encrypted evidence is correlated into a **knowledge graph** - a signed, searchable audit trail that links artifacts to identities, tools, and actions. This provides explainable line-of-sight from code to cloud.

- **Policy-as-code and guardrails**
  Policies enforce "what good looks like" (e.g., verified provenance, approved bases, SBOM present, scanners completed, zero critical vulns, signed checkpoints). Violations can block promotion or trigger workflows.

- **Runtime admission control and trust propagation**
  Integrity and policy decisions propagate to deployment - admission controllers verify signatures and provenance, so only compliant artifacts run.

- **Frictionless for developers**
  Checks run "invisibly" in existing pipelines. Developers see clear, contextual feedback and can auto-open tickets or apply safe fixes without leaving their tools.

This foundation made Scribe a strong continuous assurance platform. The next step, **and the focus of this paper**, is how Scribe evolved to make **AI the engine** behind prioritization, explainability, and remediation.

---

3) The AI-Era Shift: From AI-Assisted to AI-Centric

Many vendors bolt AI onto dashboards. Scribe took a different path: **put AI at the center** by giving it what most systems lack - **reliable, signed, contextual data**.

3.1 A Data Advantage AI Can Trust

AI is only as strong as the signals it sees. Scribe generates and gathers unique, high-fidelity signals from the SDLC:

- **Hashes, signatures, and provenance** for every artifact and stage.
- **SBOMs and AI-BOMs** that describe what software and models are actually used.
- **Scanner outputs** (SAST/SCA/DAST/secret, container, posture) normalized and signed.
- **Pipeline and infra posture** (who ran what, in which environment, with which configs).
- **Human and machine identities, approvals, and policy decisions**.

These signals flow into the **Software Knowledge Graph**, giving AI the context to answer “What is the root cause?”, “What is the blast radius?”, “Is this risk exploitable here?”, “What SDLC policy was violated and by who?”, and “Can we apply a safe, verified remediation now?”

3.2 Agentic Architecture: A Network of Specialists

ScribeAI orchestrates **purpose-built networks of agents** that collaborate based on the knowledge graph:

- **Heyman – AppSec copilot**
  Conversational interface that understands risk in context, explains root cause, triages findings, opens tickets, and coordinates workflows with humans in the loop.
- **Remus – Auto-remediation agentic workflow**
  Generates safe patches for code and configurations, validates changes, and proposes PRs/MRs; updates attestations and risk posture automatically.
- **Docktor – Docker fixer**
  Analyzes Dockerfiles for vulnerabilities and inefficiencies, suggests secure base images, reduces image size, re-evaluates the new build, and produces a report.
- **Compy – Compliance agentic workflow**
  Continuously evaluates evidence against standards (SLSA/SSDF/SAMM, FedRAMP,

DORA, PCI, CRA), highlights gaps, and drafts audit-ready artifacts.

- **Eva – Evidence agentic workflow**
  Instruments sensors and gates across the SDLC; ensures the right evidence is captured and signed at the right time.

These agents work from a single source of truth- the signed knowledge graph - so decisions are explainable and repeatable.

---

4) What Changes for Security and Engineering Leaders

4.1 From Alert Overload to Actionable Priorities

Scribe's AI considers exploitability (EPSS), reachability, asset value, environment posture, and tampering signals (provenance, signatures) to promote the **small set of issues that matter now**. Heyman explains **why** in plain language; Remus can propose fixes immediately and submit PRs to your Git.

4.2 Remediation at Developer Speed

Auto-generated PRs/MRs, verified by policy and signed checkpoints, move risk down without detours. Remus and Docktor accelerate recurring fixes (dependency upgrades, base hardening, misconfig corrections) while respecting guardrails.

4.3 Compliance Becomes Continuous

Compy maps evidence to frameworks as work happens, not at quarter-end. Attestations and reports are **always current**, and based on hard evidence from the pipelines, cutting audit prep from weeks to hours and eliminating spreadsheet drift.

4.4 Zero-Trust Starts at Commit

By enforcing signed provenance and policy at each stage, Scribe prevents untrusted artifacts from ever reaching deployment. Admission control consumes the same evidence, so "what you built" equals "what you run."

---

5) Architecture Overview

1. **Collectors & Integrations**
   Connect to SCMs (GitHub, GitLab, Bitbucket, Azure DevOps), CI/CD (Actions, GitLab CI, Jenkins, CircleCI, Argo, etc.), registries, and clusters. Over 180 integrations normalize evidence.

2. **Secure Evidence Pipeline**
   Evidence (not code) is encrypted in transit, signed, and stored in a **tamper-proof repository**. Each item is traceable to identities, tools, and times.

3. **Knowledge Graph**
   A graph database correlates artifacts, commits, SBOM entries, policies, scanner findings, approvals, and runtime objects. This is the substrate for explainable AI.

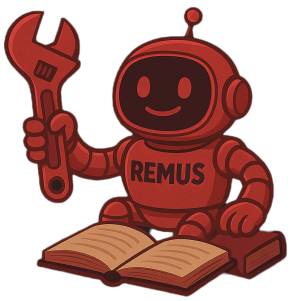
4. **Policy-as-Code**
   Customers express controls as code: required attestations, approved bases, vuln budgets, segregation of duties, release criteria. Policies gate pipelines and admission.

5. **Agent Orchestrator (ScribeAI)**
   Routes tasks (triage, analysis, fix generation, compliance mapping) to the right agent; preserves auditability; aligns agent actions with policy and human approvals.

6. **User Experience**

   - **ScribeHub dashboard** for unified visibility, lineage, compliance, and KPIs. Many of the AI-agentic workflows are managed and consumed through ScribeHub.

   - **Conversational UX (Heyman)** for natural-language queries and actions: “Show exploitable vulns in payments service,” “Open fix tickets for criticals with reachability,” “Regenerate provenance for v1.4 builds.”



6) Use Cases and Outcomes

6.1 Secure Build Provenance & Anti-Tampering

Problem: CI scripts, keys, and bases drift; attackers target build systems.
Scribe: Enforce signed provenance and in-toto attestations; verify signatures at each step; block unsigned or tampered artifacts.
Outcome: Only trusted artifacts progress and deploy; blast radius of key or script compromise is drastically reduced.

6.2 Third-Party and AI-Generated Code Transparency

Problem: Outsourced components and AI outputs ship quickly but may hide misconfigurations, secrets, or vulnerable code.
Scribe: Generate SBOMs/AI-BOMs automatically, map provenance, collect scanners' results from SARIF files or APIs, and enforce policy gates; Remus proposes fixes as PRs.
Outcome: Faster acceptance with confidence; consistent standards across internal and external code.

6.3 Continuous Compliance (SLSA, SSDF, DORA, FedRAMP, CRA)

Problem: Evidence collection is manual; audits are time-consuming and error-prone.
Scribe: Compy maps signed evidence to control catalogs; creates audit-ready reports per build; maintains an immutable trail.
Outcome: Reduce audit prep from weeks to hours; demonstrable “secure-by-design” posture.

6.4 Incident Response & Exposure Analysis

Problem: When a zero-day hits, teams can’t quickly answer “What’s affected?”
Scribe: Query the knowledge graph to locate vulnerable versions, builds, deployments, and dependents; Remus and Docktor propose remediations.
Outcome: Confident, surgical response; minimized downtime and risk.

6.5 Admission Control and Runtime Trust

Problem: Deployments accept artifacts lacking evidence; drift sneaks in between build and run.
Scribe: Admission controllers verify signatures, provenance, and policy compliance at deployment time.
Outcome: Run only what was verified; developers keep shipping, security keeps control.

---

7) Scribe vs. DevOps Platforms

DevOps suites (e.g., GitHub, GitLab) provide excellent SCM/CI and integrate scanners, but they are not purpose-built for **end-to-end, attestation-driven supply chain assurance**:

- No unified, cross-tool **integrity graph** linking code, artifacts, policies, and runtime.
- Limited **admission enforcement** based on evidence from outside their ecosystem.
- Compliance reporting is often **repo-scoped** and **manual** to aggregate.
- Remediation is typically **developer-driven**, not **agent-driven,** with auditability.

Scribe complements these platforms by discovering assets across heterogeneous estates, enforcing policy-as-code, generating and validating signed evidence, and driving **AI-powered remediation** - all with a tamper-proof audit trail, based on signed, machine-readable attestations.

---

8) Data Protection and Privacy

- **Evidence-only:** Scribe collects metadata and results; **not source code**.
- **Encryption & signing:** Data is encrypted in transit and at rest; items are signed with enterprise PKI or Sigstore.
- **Fine-grained access:** Least-privilege controls; sensitive attestations shared on a need-to-know basis.
- **Auditability:** Every agent action, policy decision, and human approval is recorded, supporting internal and external audits.

9) KPIs and Business Impact

Security leaders and platform teams typically measure success along these lines:

- **Vulnerability noise reduction:** 40–70% drop in non-actionable alerts via contextual triage (reachability, EPSS, provenance).
- **Remediation velocity:** PR-based auto-fixes reduce MTTR from weeks to days or hours for recurring findings.
- **Compliance effort:** Audit prep time reduced by >60% via continuously generated, signed evidence and reports.
- **Release stability:** Fewer late-stage blocks due to earlier, automated gates; fewer hotfixes post-release.
- **Incident response:** Accelerated impact analysis and targeted remediation through querying the attestation lake and knowledge graph.
- **Developer experience:** Less manual security toil; fixes arrive as clear PRs with rationale; predictable release cadence.

---

10) Differentiation Summary

- **Evidence-first architecture:** Signed, verifiable attestations across the SDLC, not just scan results.
- **Knowledge graph:** A tamper-proof, explainable lineage map from code to cloud; the substrate AI needs.
- **Agentic remediation:** Heyman, Remus, Docktor, Compy, Eva, - specialized virtual teams of agents that act, not just advise.
- **Heterogeneous by design:** Works across GitHub/GitLab/Bitbucket/Azure DevOps; supports diverse CI/CD and cloud stacks.
- **Developer-friendly:** Invisible checks, clear feedback, policy-aligned PRs/MRs; velocity preserved.
- **Compliance on autopilot:** Out of the box policy bundles that manifest known regulation and best frameworks. Evidence collected as work happens; reports always current.

Everything is signed and archived.

Trust at the Speed of Software

Security can no longer be a late gate or a quarterly audit. In the AI era, when code and risk scale together, organizations need **continuous, evidence-backed assurance** and **agentic remediation** that keeps pace with development.

Scribe Security has evolved from a powerful attestation platform to an **AI-centric assurance and remediation system**. By combining a signed knowledge graph with a network of intelligent agents, Scribe enables teams to **decide faster, fix sooner, prove continuously, and ship with confidence**, whether code is written by humans, generated by AI, or assembled from third-party components.

If you're ready to move from chasing vulnerabilities to **engineering trust into every release**, Scribe is built for you.

Next Steps

- See ScribeHub and Heyman in action with a targeted demo.
- Pick one service and one compliance target; measure results in 30 days.
- Expand policies and agentic workflows; turn continuous assurance and secure-by-design into your default.

Contact: info@scribesecurity.com | scribesecurity.com