A Use Case of Using Scribe Trust Hub

How Scribe Security Can Help You Manage and Control Open Source Risk in Your Supply Chain

TABLE OF CONTENTS

INTRODUCTION

Scribe is a security platform designed to minimize users' risk from their software supply chain.

As a software producer, you can use Scribe to apply security guardrails to your SDLC based on security posture, risk analysis of open-source dependencies, and development practices. You can also use Scribe to share with your customers select attestations about your compliance and software's security.

As a software consumer, you can apply guardrails to manage risk from 3rd party's open-source dependencies or entire products, consume SBOMs, and track vulnerabilities from development to post-deployment.

To instate those guardrails, Scribe collects evidence from the SDLC, enriches it with external intelligence, and analyzes it for risk. It can consume a wide range of sources and output a plethora of formats and artifacts.

These signed evidence-based artifacts become a means of transparency and a common source of truth for stakeholders: dev teams, security teams, GRC and legal, users, and customers.

A strong reporting system and analitycs capabilities allows for a quick identification of issues and support informed decision making.

COLLECTING EXTERNAL SOURCES

Scribe integrates with data sources to track and analyze risks in open-source dependencies. These integrations offer up-to-date intelligence. Some of these sources include:

1 National Vulnerability Database (NVD): provides a range of information about reported vulnerabilities in software components.

2 Vulnerability information aggregated from various data sources such as:

- GitHub Advisory Database
- National Vulnerability Database (NVD)
- PyPI Advisory Database
- Go Vulnerability Database
- Rust Advisory Database
- Global Security Database
- OSS-Fuzz
- Rocky Linux
- AlmaLinux
- Haskell Security Advisories
- RConsortium Advisory Database
- Python Software Foundation Database
- Debian sources
- Red Hat sources
- Alpine sources
- Ubuntu Oval database
- Debian Security Tracker
- Red Hat Enterprise Linux (RHEL) Oval database
- SUSE Oval database
- Oracle Oval database
- Alpine SecDB database
- VMware Photon OS database
- Amazon Web Services (AWS) UpdateInfo
- Open Source Vulnerability (OSV) Database

3 Other types of risk intelligence (reputation, licensing, aged versions)

- Deps.dev
- OpenSSF scorecard
- SPDX.org

COLLECTING AND PROCESSING SDLC SECURITY EVIDENCE

To secure the supply chain, Scribe continuously generates and collects the following types of evidence. For instance, on every build run:

1. Software bills of materials of assets and artifacts such as source code, package managers, build artifacts, and build agents
2. Hash values of artifacts and tools in the SDLC toolchain
3. Findings from scans for vulnerabilities
4. Security-related settings from dev tools
5. Information about SDLC events such as code commits, user IDs, code reviews

Scribe collects this evidence by utilizing dev tools'l APIs, CI plugins (for example, Scribe GitHub Actions), Scribe's agent - Valint, and deployment gates (for example, a K8S admission controller).

The evidence is cryptographically signed using PKI or Sigstore and validated to ensure against tampering, track back components' provenance, and to serve as an irrefutable source of truth.

The evidence is then processed into a knowledge graph that is enriched with external information about vulnerability risk and exploitability scores, as well as open source risks such as aging, reputation scores, and licensing information.

USING THE KNOWLEDGE GRAPH

Users can query the knowledge graph through a reporting interface for preventive action, such as addressing detected vulnerabilities, and for incident response, such as mapping projects affected by a newly discovered vulnerability or security incidents (e.g. a compromised CI tool, malicious repository, exploited machine, rogue developer etc.).

The Scribe policy engine uses the knowledge graph to apply guardrails through KPI measurement or enforcement.

THE ROLE OF APPLICATION SECURITY FINDINGS

For the best results in software composition analysis (SCA), we recommend using Scribe to sample different stages in the SDLC. For instance, direct dependencies are best revealed by analyzing source code, while indirect dependencies can be detected post-build.

Similarly, it is useful to fuse findings from different SCA scanners to create better coverage.

To this end, Scribe can utilize the findings from your existing SCA tools.

Scribe has a rich built-in capability to perform software composition analysis (SCA), generate SBOM artifacts, and detect vulnerabilities associated with the components before and post-deployment.

Nonetheless, there are different best-of-breed tools in the market that have particular strengths both in composition analysis and vulnerability detection—for instance, binary firmware analysis tools.

Scribe is capable of ingesting findings from different SCA tools and different steps in the SDLC. It ingests SBOMs and vulnerabilities into its knowledge graph and groups together associated findings. This way, you can utilize consolidated vulnerability reports of your tools with Scribe's additional findings.

Within Scribe, you can utilize reports by SBOM and findings grouped by tool, by SDLC step, by library (component), or by vulnerability. These reports can span your entire software portfolio.

CONCLUSION

An important control, among several additional ones, for securing the software supply chain is the composition analysis and detection of risks in open-source dependencies. Scribe leverages and enhances your existing controls by treating their findings as evidence and utilizing them for risk analysis, risk monitoring, application of SDLC guardrails, complying with best practices, and resolution.

In addition, you can review within the Scribe console the findings in different slices and aggregation levels to effectively manage, control and mitigate the risk embedded in your software products.