

# How to utilize Scribe Security to address SSDF requirements



# Table of Contents

**PAGE 3** Introduction

---

**PAGE 4** Protect the Software (PS)

---

**PAGE 6** Produce Well-Secured Software (PW)

---

**PAGE 7** Respond to Vulnerabilities (RV)

---

**PAGE 8** Prepare the Organization (PO)

---

**PAGE 9** Conclusion

---

# Introduction

The Secure Software Development Framework (SSDF), AKA NIST SP800-218, is a set of guidelines developed by NIST in response to Executive Order 14028, which focuses on enhancing the cybersecurity posture of the United States, particularly concerning software supply chain security.

SSDF is a best practices framework, not a standard. While particularly relevant to organizations that develop software for the

US government, SSDF benefits any software development organization.

This paper reviews the four SSDF practices and explains how to utilize Scribe Security to help implement them by providing tools to enable people and processes to uphold security standards in the SDLC.

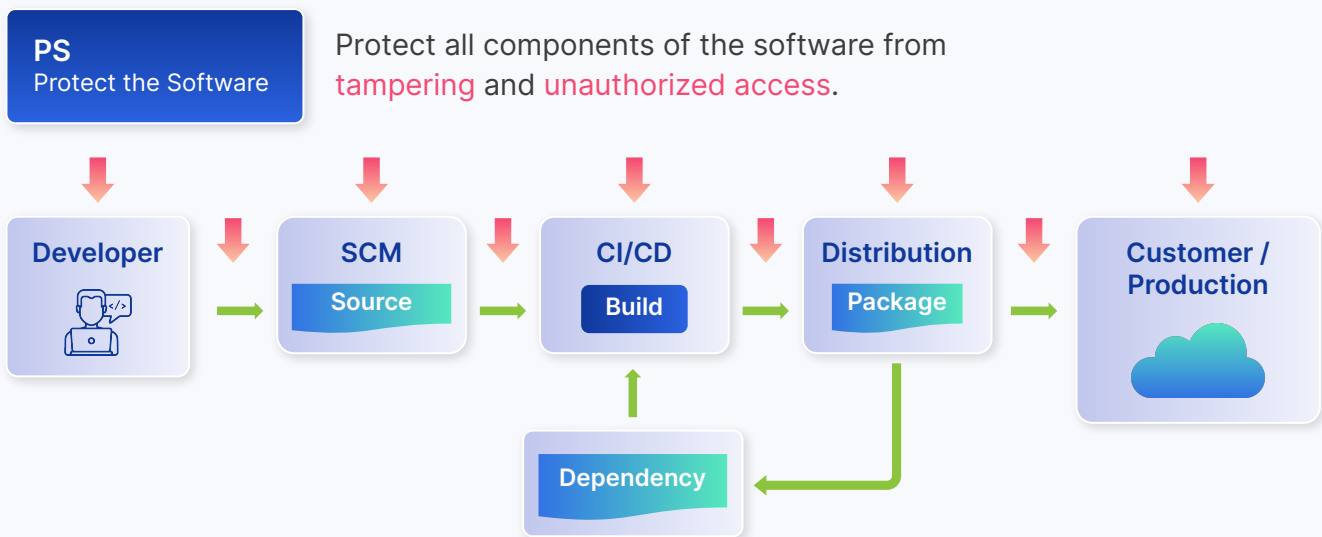
Scribe Security is a platform for securing software supply chains.

# Protect the Software (PS)






The PS practice addresses the threat to all software components of tampering and unauthorized access. PS controls aim to ensure the security of software throughout its

development lifecycle. These include protecting all forms of code, verifying software release integrity, and archiving and protecting each software release.



Scribe helps address PS controls by the following features:

-  **Signing Source Code and Build Artifacts:** This helps ensure the integrity and authenticity of the code and build artifacts, aligning with the SSDF's emphasis on protecting code from unauthorized access and tampering. Continuous code signing throughout the development process helps ensure that the integrity of software releases has not been tampered with.
-  **Tracking Provenance:** Monitoring the origin and history of software components for secure sourcing, helping to safeguard the software against vulnerabilities that arise from compromised components.
-  **Monitoring Software Integrity During SDLC:** Continuous checks for integrity throughout the SDLC impede potential attackers in the dev tools and pipelines.

4



**Automated Security Toolchains:** Implementing toolchains that automate aspects of the security process can reduce human error and ensure consistent application of security practices.

5



**Criteria for Software Security Checks:** Defining and using criteria to check software's security during its development is crucial for ensuring compliance with organizational standards and identifying potential vulnerabilities.

6



**Secure Development Environments:** Maintaining secure development, build, and test environments is vital to prevent vulnerabilities and safeguard the software development process.

7



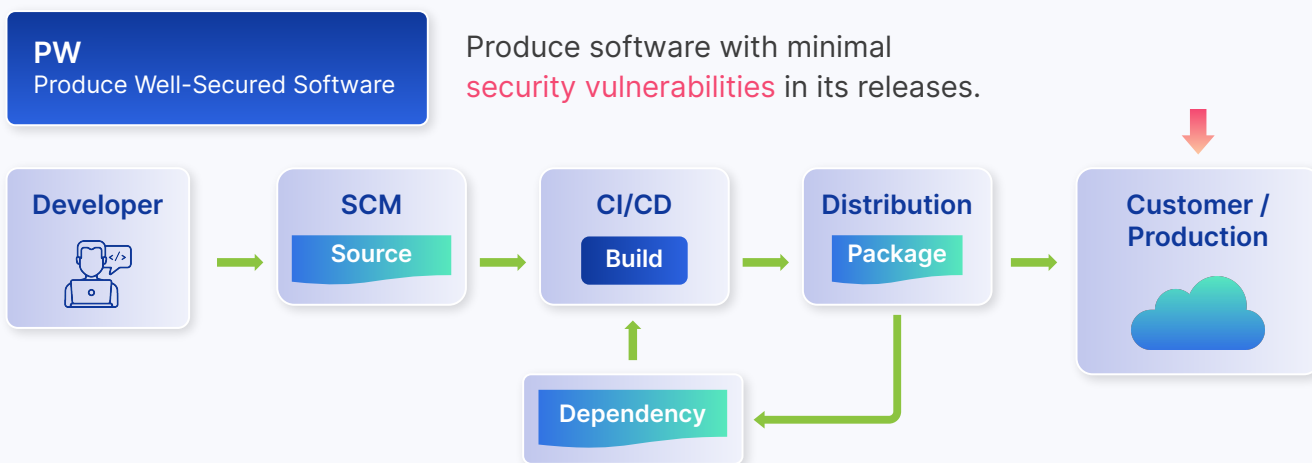
**Apply policy to access controls to source code, dev tools, and software artifact registries:** Restricting access to authorized users can prevent unauthorized use and tampering.

# Produce Well-Secured Software (PW)








The PW practice addresses vulnerabilities introduced into the software throughout the SDLC. It includes designing software to meet security requirements, reviewing designs for

compliance, reusing secure code, configuring build processes for security, and testing executable code for vulnerabilities.



Scribe helps address PW controls through the following features:

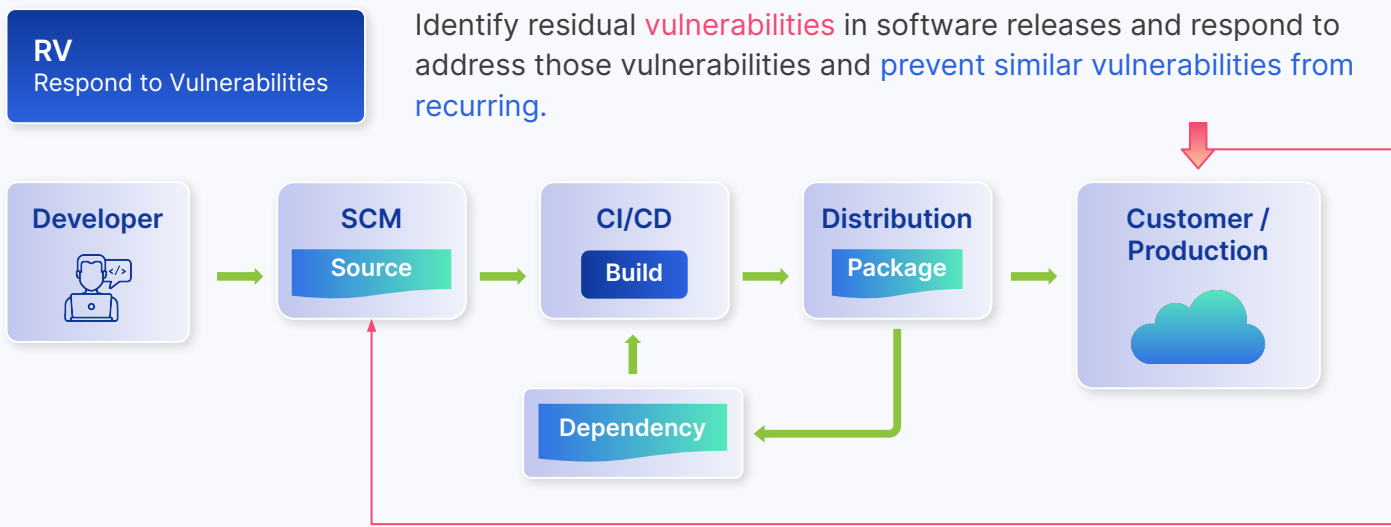
-  **1 Software Composition Analysis (SCA):** Identifying and managing open-source and third-party components to mitigate risks associated with software dependencies.
-  **2 Vulnerability and Reputation Intelligence:** Continuous monitoring for vulnerabilities and assessing the reputation of software components to ensure reliability.
-  **3 Risk Analysis:** Evaluating and monitoring potential security and license risks associated with open-source dependencies, managing Software Bill of Materials (SBOMs) and Vulnerability Exploitability eXchange (VEX) advisories.
-  **4 Security-Related Evidence Gathering from SDLC:** Analyzing data throughout the SDLC for informed risk management.
-  **5 Secure Development Policies:** Codifying guidelines for software development processes and procedures and measuring KPIs across teams and software development projects.

# Respond to Vulnerabilities (RV)







The RV practice involves identifying residual vulnerabilities in software releases and responding appropriately. RV involves either

resolving vulnerabilities, mitigating, or providing stakeholders with advisories.



Scribe helps address RV controls through the following features:

- 1**  **Gathering Evidence from SDLC:** Collecting metadata throughout the SDLC that creates a snapshot of all the assets, tools, dependencies, and process steps for every software release across the organization.
- 2**  **Knowledge Graph for SDLC Intelligence:** enabling analysis of the impact of new vulnerabilities, problematic dependencies, or problematic code contributors across the entire project portfolio using knowledge graphs.
- 3**  **Ongoing Vulnerability Identification and Prioritization:** automated tools for identifying, and prioritizing vulnerabilities.
- 4**  **Continuous Monitoring and Improvement:** continuous monitoring mechanisms to detect newly published vulnerabilities post-release across the organization's software portfolio.

# Prepare the Organization (PO)

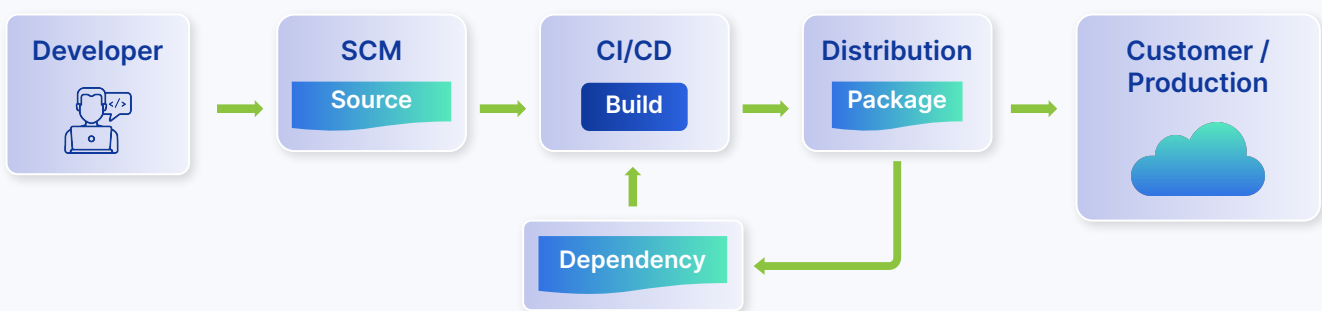


The PO practice aims at the organizational level. People, processes, and technology are





prepared to perform secure software development.

**PO**  
Prepare the Organization

Ensure organization's people, processes, and technology perform secure software development at org level and dev group/ project level



Scribe helps address PO controls through the following features:

-  **1 Policy as Code (GitOps):** Implementing security policies as code allows for consistent guardrails for security throughout the SDLC. This aligns with the SSDF's emphasis on embedding security into every stage of software development.
-  **2 Compliance with Frameworks like SLSA, SSDF PS, and Custom Initiatives:** Enforcing compliance with these frameworks ensures that software development aligns with recognized security standards, further solidifying the organization's commitment to secure software development.
-  **3 Enhanced Communication:** Facilitating effective communication and collaboration among teams based on real evidence data and KPIs as a common ground truth to ensure security requirements are well-understood and integrated into the development process.
-  **4 Continuous Monitoring and Auditing:** continuous monitoring and auditing mechanisms to ensure ongoing adherence to security practices and policies.



# Conclusion

Scribe Security is a platform for securing software supply chains. It is designed to address all the security aspects of protecting the software artifacts and the software factory.

It allows comprehensive and holistic risk management of software products across and between organizations and can act as a

software trust center between software producers and consumers.

Scribe is also a means of compliance and **supports the SSDF framework**. It allows compliance with other best practices such as SLSA, specific industry standards, and custom organizations initiatives.

**PO**

SDLC Guardrails, Compliance (SSDF, SLSA, SBOM...)

**PS**

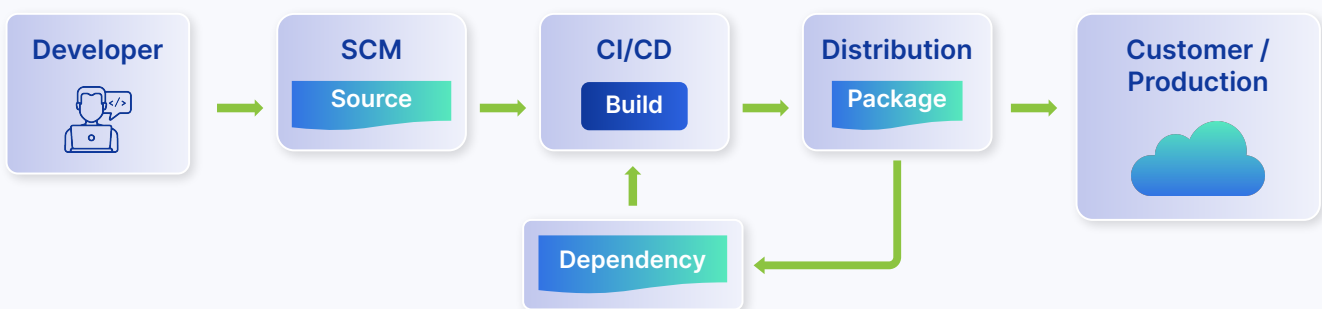
Code & Artifact Signing, Provenance, SW Integrity

**PW**

SCA, Vuln & Reputation Intel, Risk Analysis, SBOM (VEX), Security Evidence

**RV**

SDLC Intelligence & Impact Analysis



If you got all the way down here,  
you must be ready to get started!

[START FOR FREE](#)

Have more questions?

[Contact Us](#)

 scribe

Want to see it in action?

[Schedule a Demo](#)