OpenSSF
OPEN SOURCE SECURITY FOUNDATION

OPENCHAIN

Strengthening Software Supply Chain Security

with OpenChain and OpenSSF



Contents

Executive Summary 3

Introduction 4

1. OpenSSF and OpenChain: Two Complementary Perspectives on Software Supply Chain Assurance 5

2. From Software Supply Chain Challenges to Complementary Solutions 8

3. Building Trustworthy Software Through Governance, Engineering, and Evidence 9

4. Supporting Manufacturers in Meeting Emerging Regulatory Expectations 11

5. A Path Forward: Adoption and Collaboration Opportunities 14

Conclusion and Next Steps 16

Executive Summary

Software supply chain assurance has become a business imperative and an area of increasing customer and regulatory attention. Modern software is increasingly assembled from internally developed code, open source software, closed source components, and third-party services. As software supply chains become more complex, organizations are expected not only to manage risks but also to demonstrate that appropriate organizational processes, engineering practices, and supporting evidence are in place. This can be especially challenging for the open source software that forms the foundation of most modern software products.

Software supply chain integrity requires more than individual security controls or management activities. It depends on integrating organizational capabilities, secure engineering practices, and evidence that these activities are implemented consistently throughout the software lifecycle. Organizational capabilities and secure engineering are mutually reinforcing elements of software supply chain security.

The OpenChain Project and the Open Source Security Foundation (OpenSSF) contribute complementary capabilities to this objective. OpenChain focuses on building the organizational capabilities needed to manage software containing open source software, and OpenSSF focuses on secure engineering practices. Together, they help organizations establish a risk-based approach to software supply chain assurance and generate reusable evidence that strengthens transparency and accountability while helping organizations meet evolving customer and regulatory expectations.

This paper explains how these approaches can be applied across the software lifecycle to strengthen software supply chain practices. It shows how governance and engineering practices reinforce one another to produce reusable evidence that supports supplier relationships, customer expectations, and evolving regulatory requirements. Adoption of OpenChain specifications or OpenSSF guidance does not, by itself, constitute compliance with any specific regulation. Together, they provide building blocks that organizations can incorporate into broader software security programs to efficiently meet many regulations and requirements.

Introduction

Modern software is rarely developed entirely from scratch. Instead, it is assembled from internally developed code, open source software, commercial components, and services, including those from cloud providers. This model has accelerated innovation over the last decade and enabled unprecedented collaboration across the software ecosystem, but it has also increased the complexity of managing software supply chain risks across multiple sources and dependencies.

Unlike proprietary software acquired from a single supplier, open source software encourages organizations to take an active role in managing the components they use. This provides opportunities beyond simply identifying vulnerabilities or choosing security tools, but to seize those opportunities organizations must coordinate governance, engineering, suppliers, and open source communities while maintaining evidence that these activities are performed consistently. Customers, business partners, and regulators increasingly expect organizations not only to implement appropriate security measures, but also to demonstrate that software supply chain risks are understood and managed systematically. Meeting these expectations requires more than individual tools or isolated security measures.

It also depends on organizational capabilities that establish clear responsibilities and repeatable processes, secure engineering practices integrated throughout the software lifecycle, and objective evidence that these activities are implemented consistently. These consumer organizations also play an important role in contributing improvements and security fixes back upstream, which helps to strengthen the resilience of the open source ecosystem on which they depend.

An organization that fails to contribute to upstream open source projects risks becoming less competitive because it must fund its work independently and may falling behind the rest of the industry that collaborates.

The OpenChain Project and OpenSSF address different but connected aspects of software supply chain assurance. This paper explores how these approaches can be combined to help organizations strengthen their practices throughout the software lifecycle, generate reusable evidence, and respond to evolving customer, business, and regulatory expectations. It also identifies opportunities for continued collaboration between the two communities to improve interoperability and simplify adoption.

1. OpenSSF and OpenChain: Two Complementary Perspectives on Software Supply Chain Assurance

OpenChain and OpenSSF address software supply chain assurance from complementary perspectives. Although they originated with different missions, both OpenChain and OpenSSF address software supply chain security. OpenChain establishes the organizational capability needed to consistently manage software containing open source software, whereas OpenSSF advances the engineering practices that strengthen software security throughout the software development lifecycle. Together, they demonstrate that effective software supply chain assurance depends on both organizational processes and secure engineering practices.

As software supply chains have become more complex, organizations increasingly need governance that establishes clear responsibilities and repeatable processes, together with engineering practices that implement those requirements throughout the software lifecycle. OpenChain and OpenSSF share the common objective of improving trust in the software organizations develop, integrate, and distribute.

OpenSSF: Advancing Secure Software Development

OpenSSF is a cross-industry initiative dedicated to improving the security of open source software through community collaboration. Its work spans secure software development, software integrity, vulnerability management, software transparency, build security, security education, AI security, and cybersecurity policy. Rather than prescribing a single implementation model, OpenSSF develops practical guidance, community specifications, and technical initiatives that organizations can adopt according to their technologies, development environments, and risk profiles.

Representative initiatives such as OpenSSF Scorecard, the Open Source Project Security (OSPS) Baseline, Security Insights, SLSA, Sigstore, OSV, and GUAC help organizations assess project security, strengthen build integrity, manage vulnerability information, establish software provenance, and correlate software supply chain metadata for risk analysis. Complementing these technical initiatives, OpenSSF working groups and special interest groups develop implementation guidance, promote interoperability, develop training and documentation, and help align engineering practices with evolving policy and regulatory expectations. Together, these activities provide organizations with practical resources and reusable technical evidence that contributes to software supply chain security and risk management.

OpenChain: Establishing Organizational Capability

OpenChain focuses on helping organizations establish consistent and repeatable organizational practices for managing software containing open source software. OpenChain specifications define the organizational capabilities needed to manage software supply chain activities effectively while allowing organizations flexibility in how those capabilities are implemented.

ISO/IEC 5230 defines the key requirements for a quality open source license compliance program, and ISO/IEC 18974 extends this approach to open source security assurance by defining the organizational capabilities needed to identify, assess, communicate, and manage known vulnerabilities in software containing open source components throughout the software lifecycle. Like ISO/IEC 5230, it specifies what capabilities an organization should establish rather than prescribing how they should be implemented.

Both ISO/IEC 5230 and ISO/IEC 18974 are intentionally lightweight and outcome-oriented, enabling organizations of different sizes and maturity levels to implement them using processes and tools appropriate to their environment.

Together, these specifications establish a framework based on documented policies, defined responsibilities, personnel competence, supplier engagement, vulnerability management processes, retained records, and periodic review. This helps organizations manage software supply chain activities consistently while continually improving their capabilities as products, technologies, and threats evolve.

Different Entry Points, Shared Objectives

The primary difference between the two lies in their point of entry.

OpenChain begins with organizational capability: Are policies, responsibilities, and processes in place? Can the organization demonstrate that they are consistently implemented?

OpenSSF begins with secure engineering: Are secure development practices implemented? Are dependencies and vulnerabilities managed? Can software integrity and provenance be demonstrated?

These perspectives are complementary rather than overlapping. Organizational processes establish the framework within which engineering practices can be implemented consistently, while engineering provides the technical controls that put those processes into practice. Neither is sufficient on its own: governance without implementation lacks operational impact, and engineering without governance can be difficult to sustain, scale, or demonstrate across the organization.

Throughout this paper, software supply chain assurance is considered across a common set of organizational and technical capabilities: secure development, lifecycle risk management, vulnerability management, software transparency, software integrity and provenance, technical documentation and records, supplier and dependency management, and the ability to demonstrate that these practices are implemented consistently. OpenChain and OpenSSF contribute to these capabilities from complementary organizational and engineering perspectives.

Areas of Collaboration and Natural Touchpoints

The relationship between OpenChain and OpenSSF becomes particularly evident when organizational and engineering activities are viewed together across the software lifecycle. OpenChain establishes the organizational framework, whereas OpenSSF offers practical mechanisms for implementing and demonstrating those processes.

Examples of these natural touchpoints include:

SOFTWARE SUPPLY CHAIN ACTIVITY	OPENCHAIN	OPENSSF
Organizational governance	Policies, roles, responsibilities, competence, documented procedures	Security guidance and community best practices
Component management	Processes for managing software inventories and supplier interactions	SBOM generation, dependency analysis, GUAC
Secure development	Governance and accountability	Secure SDLC guidance, focused security technical guidance, baseline, SLSA, OpenSSF Scorecard
Software integrity	Release and verification processes	Sigstore, provenance, build integrity
Education/training	Educational courses on licensing	Educational courses on secure software development
Vulnerability management	Defined processes and supplier communication	OSV, OpenVEX, vulnerability guidance, security tooling
Assurance evidence	Policies, records, audit evidence	Machine-readable artifacts such as SBOMs, provenance, scorecards, badges, security assessments

The examples above illustrate that OpenChain and OpenSSF address different but connected aspects of software supply chain assurance. OpenChain establishes the organizational capability needed to consistently manage software supply chain activities. OpenSSF provides the technical practices and artifacts that improve implementation. Used together, they enable organizations to connect governance with engineering across the software lifecycle.

2. From Software Supply Chain Challenges to Complementary Solutions

Modern software is assembled from internally developed code, open source software, commercial components, cloud services, and third-party dependencies. While this model accelerates innovation, it also increases the complexity of managing software supply chain risks.

These risks extend beyond technical vulnerabilities. Organizations must establish clear organizational responsibilities and repeatable management practices in parallel with embedding secure engineering practices into everyday development. Across industries, customers, business partners, and regulators also expect organizations to demonstrate these practices through objective evidence rather than assertions.

Common Software Supply Chain Challenges

Organizations face a common set of software supply chain challenges throughout the software lifecycle:

UNDERSTANDING SOFTWARE COMPOSITION. Organizations need visibility into the software they develop and distribute, including open source software, commercial components, and transitive dependencies. Maintaining accurate software inventories and SBOMs supports vulnerability management, license compliance, software transparency, and communication across the software supply chain.

MANAGING VULNERABILITIES. New vulnerabilities emerge continuously. Organizations must identify affected components, assess risk, prioritize remediation, coordinate with commercial suppliers and upstream open source communities where appropriate, and document the actions taken.

MANAGING SOFTWARE ECOSYSTEMS. Modern software depends on commercial suppliers, service providers, and open source communities. Unlike commercial suppliers, open source communities have no contractual obligations toward downstream users. Organizations therefore remain responsible for evaluating, integrating, maintaining, and updating the open source software they choose to use, while contributing improvements upstream whenever possible.

EMBEDDING SECURITY THROUGHOUT DEVELOPMENT. Security is most effective when integrated throughout the software lifecycle through secure design, dependency management, build integrity, software provenance, automated security testing, and software transparency.

COORDINATING ORGANIZATIONAL RESPONSIBILITIES. Software supply chain assurance involves engineering, security, legal, procurement, quality assurance, and product management. Clearly assigned responsibilities and repeatable practices help organizations perform these activities consistently and generate reusable evidence.

Complementary Solutions

No single practice addresses all of these challenges. Organizations need both organizational capabilities and secure engineering practices to manage software supply chain risks consistently and demonstrate that appropriate measures have been implemented.

OpenChain helps organizations establish the organizational foundation for managing software supply chain activities, while OpenSSF delivers practical engineering guidance, community initiatives, and technical projects that strengthen software security throughout the lifecycle. The next chapter examines how these capabilities work together to produce reusable assurance evidence.

3. Building Trustworthy Software Through Governance, Engineering, and Evidence

Software supply chain assurance is not achieved through a single activity or deliverable. It results from the consistent application of organizational capabilities, secure engineering, and software supply chain management throughout the software lifecycle. Organizations increasingly need to demonstrate not only that appropriate security measures have been implemented, but also that these measures are reinforced by defined processes, assigned responsibilities, and objective information.

Governance, Engineering, and Assurance Artifacts

Software security is most effective when governance and engineering reinforce one another. Policies, defined responsibilities, and repeatable processes provide the organizational foundation, complemented by engineering practices that implement them through secure development, build integrity, vulnerability management, software transparency, and automation.

This integration extends beyond software development teams. Engineering, security, legal, procurement, quality assurance, and product management all contribute to managing software supply chain risks. Mature software supply chain management therefore depends on cross-functional collaboration enhanced by clearly defined responsibilities.

Equally important is the ability to generate machine-readable artifacts as a natural outcome of these activities. Rather than existing solely for audits or regulatory assessments, evidence becomes an operational asset that supports decision-making, supplier communication, incident response, and continuous improvement. For that evidence to remain an operational asset, organizations also need clear ownership and proportionate arrangements for its retention and retrieval.

Demonstrating Software Supply Chain Assurance

The specific signals and records organizations maintain will vary according to their products, contractual obligations, and regulatory environment. In practice, however, organizations increasingly need to demonstrate a common set of organizational and technical capabilities:

- **Governance:** documented policies, defined responsibilities, competence, supplier engagement, and repeatable processes.
- **Secure engineering:** secure development practices, dependency management, build integrity, software provenance, code review, and security testing.
- **Vulnerability management:** processes and records demonstrating that vulnerabilities are identified, assessed, prioritized, communicated, and remediated.
- **Software transparency:** machine-readable artifacts such as SBOMs, scorecards, badges, security assessments, software inventories, provenance information, and related artifacts describing software composition and integrity.

Together, these organizational records and technical artifacts provide the information needed to demonstrate that software supply chain risks are being managed systematically throughout the software lifecycle.

Reusable Assurance Information

The concept of due diligence is increasingly reflected in cybersecurity legislation and guidance, although its precise implementation continues to evolve across jurisdictions. In this paper, due diligence is used to describe the continuous process of identifying, assessing, managing, and documenting software supply chain risks through governance, secure engineering practices, supporting documentation, and machine-readable artifacts.

Because software, dependencies, threats, and customer expectations continually evolve, organizations need to identify risks, implement proportionate controls, monitor change, and maintain evidence over time rather than treating assurance as a point-in-time activity.

One of the key advantages of combining OpenChain and OpenSSF practices is that governance and engineering activities naturally produce reusable assurance artifacts. Policies, review records, SBOMs, provenance, vulnerability information, machine-readable security attestations, and other artifacts can and should support operational decision-making, supplier communication, customer requests, and regulatory activities without being recreated for each purpose. A reusable assurance document is valid only when tied to a specific question, exact software version, and timestamp. Without all three explicitly displayed, the artifact cannot establish context for downstream auditors or customers.

These artifacts and metadata are widely interconnected. Projects such as GUAC illustrate how SBOMs, provenance, vulnerability information, security assessments, and related metadata can be correlated to advance continuous, ecosystem-scale risk analysis. This reflects a broader shift in software supply chain assurance: due diligence depends on continuously generated, reusable signals rather than static documentation.

Importantly, artifacts alone do not demonstrate that software is secure or compliant with a particular regulation. Their value lies in demonstrating a systematic, risk-based approach to managing software supply chain risks.

4. Supporting Manufacturers in Meeting Emerging Regulatory Expectations

Organizations that develop, integrate, or distribute software face growing expectations from regulators, customers, and business partners to demonstrate that software supply chain risks are managed throughout the software lifecycle.

The Cyber Resilience Act (CRA) as Part of a Broader Trend

The European Union's CRA illustrates this broader shift by establishing lifecycle cybersecurity requirements for products with digital elements, including secure development, secure-by-default, vulnerability handling, ongoing maintenance, and security updates.

The CRA's lifecycle perspective aligns well with the respective roles of OpenChain and OpenSSF. ISO/IEC 18974 defines the organizational capabilities needed to implement higher-level management of software security throughout the product lifecycle, whereas OpenSSF provides engineering guidance, technical initiatives, and community best practices that operationalize the relevant software supply chain assurance capabilities across the secure software development lifecycle.

The NIS2 Directive provides a related perspective requiring essential and important entities to implement appropriate cybersecurity risk-management measures, including supply chain security and the security-related aspects of relationships with suppliers and service providers. Although NIS2 operates at the organizational level rather than the product level, it reinforces the same emphasis on systematic software supply chain risk management.

Although the CRA only applies within the European Union, its underlying principles extend well beyond a single jurisdiction. Similar expectations are reflected in procurement requirements, industry frameworks, national cybersecurity strategies, and customer demands. Organizations are increasingly expected to demonstrate visibility into software components, secure development practices, vulnerability management, software transparency, and the ability to maintain products securely throughout their supported lifetime.

From Requirements to Organizational Capability

As these expectations evolve, organizations need confidence that software supply chain activities are performed consistently. Depending on their products and obligations, this may include implementing and maintaining the software supply chain capabilities described earlier, together with the records and artifacts needed to demonstrate their implementation.

Rather than producing documentation solely for audits, organizations benefit from generating these artifacts and records as a natural outcome of their development, management, and engineering processes, allowing these materials to be reused across customer requests, procurement, supplier relationships, and regulatory assessments.

How OpenChain and OpenSSF Support Regulatory Readiness

The expectations reflected in the CRA and similar initiatives are increasingly outcome-oriented. Rather than prescribing specific technologies or implementation methods, they emphasize the software supply chain assurance capabilities described earlier in this paper.

This outcome-oriented approach is also reflected in broader European cybersecurity legislation. For example, Article 21(2)(d) of the NIS2 Directive requires essential and important entities to address supply chain security, including the security-related aspects of relationships with direct suppliers and service providers.

As illustrated in Chapter 1 of this document, these respective capabilities span various supply chain assurance activities. OpenChain provides the organizational structure for managing security-related activities consistently over time, while OpenSSF offers practical mechanisms for implementing secure development, software integrity, vulnerability management, and transparency throughout the software development lifecycle. In the context of emerging regulatory expectations, these combined capabilities become particularly relevant because they help organizations translate broad regulatory objectives into practical organizational and technical practices.

OpenChain focuses on the organizational capability needed to manage software supply chains through documented policies, defined responsibilities, competence, supplier engagement, and repeatable processes. OpenSSF complements this with practical engineering guidance, community specifications, and technical initiatives that strengthen secure development, software integrity, software transparency, vulnerability management, and build security.

For example, OpenChain establishes organizational processes for managing software inventories and supplier interactions, and OpenSSF provides projects such as SBOM manipulation, dependency analysis, and GUAC to improve visibility into software composition. OpenChain defines organizational accountability for secure development and release activities, while OpenSSF develops practical guidance through initiatives such as SLSA, OpenSSF Scorecard, and Sigstore to strengthen secure development, build integrity, and software provenance. Similarly, OpenChain establishes repeatable processes for vulnerability handling and communication, while OpenSSF contributes to those activities through OSV, OpenVEX, and related guidance for sharing and consuming vulnerability information. Together, these organizational and technical capabilities produce machine-readable signals that organizations can reuse to support customer assurance, procurement, and regulatory activities.

This relationship closely mirrors the natural touchpoints exemplified earlier in this paper. Organizational capabilities establish how software supply chain activities are managed and are complemented by engineering practices that determine how those activities are implemented and evidenced. Combined, they help organizations produce the technical and organizational information increasingly expected to support customer demands, procurement activities, conformity assessments, and other regulatory obligations.

Although concepts such as appropriate cybersecurity measures and due diligence continue to evolve through legislation, standards, and regulatory guidance, organizations that integrate these complementary capabilities are generally better positioned to demonstrate a systematic and risk-based approach to software supply chain assurance.

Supporting (Not Replacing) Compliance

OpenChain specifications and OpenSSF guidance are not compliance frameworks and do not, by themselves, demonstrate conformity with legal or contractual requirements. Compliance depends on the specific obligations applicable to an organization's products and markets.

Instead, they provide complementary governance and engineering building blocks that help organizations establish mature software supply chain assurance programs while generating reusable records and evidence that can support customer, supplier, and regulatory expectations.



5. A Path Forward: Adoption and Collaboration Opportunities

As software supply chains continue to evolve, organizations benefit from approaches that integrate organizational capability with secure engineering throughout the software lifecycle. Rather than treating these as separate initiatives, they can adopt complementary practices incrementally, according to their products, technologies, and risk profile.

There is no single implementation path. OpenChain promotes this incremental approach through freely available self-certification checklists for ISO/IEC 5230 and ISO/IEC 18974, enabling organizations to assess existing capabilities, identify opportunities for improvement, and progressively build conformance over time.

For some organizations, the next step may be defining policies, roles, and responsibilities for managing software containing open source software. Others may prioritize strengthening secure development, software integrity, vulnerability management, SBOM management, provenance, or supplier communication. OpenChain and the OpenSSF ecosystem provide flexible resources that allow organizations to improve progressively while building artifacts that can be reused across customer, supplier, and regulatory interactions.



Collaboration Opportunities

The complementary relationship between OpenChain and OpenSSF also creates opportunities for closer collaboration between the two communities. By aligning organizational guidance with practical engineering resources, they can help organizations adopt software supply chain practices more consistently while reducing duplication and improving interoperability.

Areas for continued collaboration include:

- shared guidance showing how organizational and engineering practices fit together across the software lifecycle;
- mappings between OpenChain specifications, OpenSSF initiatives, and widely adopted standards, frameworks, and regulatory expectations;
- reference implementations combining organizational processes with secure development workflows, SBOM management, provenance, and automation;
- common terminology for software supply chain concepts; and
- interoperable models that enable policies, SBOMs, provenance, vulnerability information, and other security artifacts to be reused across multiple activities.

Equally important is the continued development of constructive liaisons between the OpenChain and OpenSSF communities. Regular coordination, joint working activities, and shared implementation guidance can help align priorities, reduce fragmentation, and provide organizations with a more coherent set of resources for improving software supply chain practices.

Looking Ahead

Software supply chain assurance is a shared responsibility involving software producers, manufacturers, commercial suppliers, open source communities, customers, and regulators. As expectations continue to evolve, organizations will increasingly benefit from approaches that combine effective organizational practices with secure engineering and transparent, reusable artifacts.

By continuing to strengthen collaboration, including formal and informal liaisons, OpenChain and OpenSSF can simplify adoption, improve interoperability, and help organizations build software supply chain practices that are practical, scalable, and adaptable to future challenges.



Conclusion and Next Steps

Software supply chains have become central to modern software development, making software supply chain assurance a strategic business capability rather than solely a technical concern. As customer expectations, industry practices, and regulatory requirements continue to evolve, organizations are expected to demonstrate that they manage software supply chain risks through effective organizational practices, secure engineering, and objective, reusable artifacts generated throughout the software lifecycle.

As this paper has shown, trustworthy software depends on combining organizational capability with technical implementation. Organizational structures establish clear responsibilities, repeatable processes, and accountability, whereas engineering practices reduce risk through secure development, vulnerability management, software integrity, and software transparency. Collectively, they provide a stronger foundation for managing software supply chain risks than either approach alone.

OpenChain and OpenSSF contribute distinct but connected capabilities toward this objective. OpenChain helps organizations establish the organizational foundations for managing software containing open source software, while OpenSSF provides community-driven guidance, projects, and best practices that strengthen software security throughout the lifecycle. The two initiatives enable organizations to adopt a practical, risk-based approach that enables both implementation and the ability to demonstrate how software supply chain risks are managed.

Looking ahead, continued collaboration between the OpenChain and OpenSSF communities offers an opportunity to simplify adoption, improve interoperability, and provide organizations with a more coherent set of resources. Shared guidance, mappings between organizational and technical practices, reference implementations, common terminology, interoperable artifacts, and constructive liaisons between the two projects can further reduce fragmentation and help organizations translate high-level objectives into practical implementation.

Ultimately, software supply chain trust is not delivered by a single standard, framework, or tool. It is built through the continuous integration of organizational capability, secure engineering, and reusable information that underpins transparency, informed decision-making, and continuous improvement. By building on the respective strengths of OpenChain and OpenSSF, and by strengthening collaboration between their communities, organizations can improve trust in the software they develop, integrate, and distribute while remaining well positioned to respond to evolving expectations.

OpenSSF
OPEN SOURCE SECURITY FOUNDATION

OPENCHAIN



openssf.org