# Comparing software supply chain protection approaches

Carlos Eduardo Miranda Zottmann [*] Thiago Melo Stuckert do Amaral [†]
Rafael Rabelo Nunes [‡] João José Costa Gondim [§]
Programa de Pós-Graduação Profissional em Engenharia Elétrica - PPEE
Departamento de Engenharia Elétrica – Universidade de Brasília (UnB)
Brasília – DF – Brasil
Email: carlos.zottmann@gmail.com[*], thiago.melo.stuckert@gmail.com[†]
rafaelrabelo@unb.br[‡], gondim@unb.br[§]

2023 Workshop on Communication Networks and Power Systems (WCNPS) | 979-8-3503-4357-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/WCNPS60622.2023.10344631

***Abstract*—Software supply chain attacks are not entirely new. Still, in recent years, a series of relevant high-impact incidents demonstrated they evolved in complexity and have brought them to scrutiny and attention. In response, several initiatives concerning protecting against the associated risks formulated frameworks and best practice guides. This study compares some of these initiatives, analyzing taxonomies, threats, and security controls. As a result, knowledge about these initiatives was organized, verifying their similarities, and discussing security controls they propose to categorize and understand them. This paper contributes to understanding the subject, suggests advances and future works, and properly identifies controls to prevent incidents.**

***Keywords*—Software supply chain, taxonomy, security controls, comparison**

## I. Introduction

Many organizations have improved their protective mechanisms to prevent cyberattacks targeting their information technology infrastructure. However, a new class of sophisticated attacks is becoming increasingly frequent and more harmful by exploiting weaknesses in software suppliers' environments [1]. In recent years, a series of cyber incidents of this nature have had a worldwide impact. Among them are the Solar Winds case [2] and the vulnerability in the Log4J open source library [3].

According to Gartner, 45% of organizations are expected to suffer an attack in their supply chain by 2030 [4]. Considering this trend, organizations with expertise in security have published frameworks and guidelines on best practices. It helps companies protect against the risks associated with the software supply chain. A review of the main frameworks and best practice guides reveals an effort to establish the attacks or threats taxonomies to the supply chain, in order to provide a better categorization and understanding of them, as well as to identify the security controls that can be applied to prevent incidents.

The aim and contribution of this study is to compare the proposed taxonomies and sets of controls in order to identify the best contributions of each. It is organized as follows: after this introduction, theoretical reference where the main frameworks and best practice guides are identified (Section II), the taxonomies covered are presented (Section III), and compared (Section IV), and discussed (Section V) with respect to the security controls they propose. Section VI sums up the discussion, presenting conclusions and final remarks.

## II. Theoretical Reference

This section introduces the concepts for a better understanding of the scenarios for exploiting vulnerabilities in the software supply chain and the existing approaches for protecting them.

### A. Key concepts

Software supply chain security has similarities with close disciplines, which must be conceptualized to correctly understand the scope of this article: The cyber supply chain, the software supply chain, the cybersecurity risks of supply chains in general, and the discipline of secure software development.

*1) Cyber supply chain:* Supply chain, in general, relates to the ecosystem of processes, people, organizations, and distributors involved in the production and delivery of a service or final product [5]. In the digital context, the cyber supply chain encompasses a wide variety of software and hardware resources, data storage, as well as artifact distribution and storage mechanisms, used to produce a digital product [1].

*2) Software suply chain:* In the context of software development, the software supply chain is the sequence of steps that are performed to create a software artifact. It is made up of source codes, builds, dependencies, and packages. An artifact's supply chain is the combination of the supply chain of its dependencies associated with its own source codes and builds [6].

*3) Cyber Supply Chain Risk Management (C-SCRM):* C-SCRM is a systematized process for managing exposure to cyber risks throughout the supply chain and developing strategies, policies, processes, and procedures to address them [7].

*4) Secure Software Development Frameworks:* Secure software development is the discipline that seeks to integrate practices aimed at producing secure software into the software development cycle (SDLC), to reduce vulnerabilities and the attack surface of the software produced. [8].

Examples of secure software development frameworks are the NIST Secure Software Development Framework (NIST SSDF)[8], Building Security in Maturity Model (BSIMM) [9], OWASP Software Assurance Maturity Model (OWASP SAMM) [10] and Microsoft Security Development Lifecycle (SDL) [11].

Such frameworks address secure software development as a whole and are frameworks with which software supply chain security concepts must be integrated.

Having clarified these concepts, the scope of this article focuses on software supply chain security. Therefore, taxonomies and security controls defined and recommended by frameworks and best practice guides regarding the secure software development or the cyber supply chain, in general, will not be considered. Only documents relating specifically to software supply chain security comprise the information that will be evaluated.

### B. Definition of software supply chain attacks

Defining what attacks on the software supply chain are is important for defining the scope of this study and its understanding. The Open Software Supply Chain Attack Reference (OSC&R) defines software supply chain attacks as any attack that targets products generated at all stages of the development cycle [12].

The Cloud Native Computing Foundation (CNCF) defines them as the compromise of components or processes used in software production, aiming to insert vulnerabilities in the developed software, with the aim of compromising the environments of end customers [13].

### C. Research method for identifying frameworks

To identify the frameworks, research was conducted on the Web of Science database and Google Scholar. The websites of large technology solution providers such as Google and Microsoft were also consulted. The searches were filtered using the keywords "software supply chain security", "taxonomy", "C-SCRM", "risks", and "vulnerabilities". And the combination of these terms using the logical AND operator. The frameworks with the most references and with the most recent publication date were considered.

## III. Taxonomies relating to the Software Supply Chain

Taxonomies can help understand both the threat landscape for the open-source software supply chain and protection approaches. Proposals for taxonomies that deal with both subjects were identified in the literature.

*1) Taxonomy of Attacks on Open-Source Software Supply Chains:* The work "Taxonomy of Attacks on Open-Source Software Supply Chains" [14] was developed with the specific objective of defining a taxonomy of attacks on the software supply chain. It was organized based on a systematic literature review, and subjected to validation by cybersecurity experts and software developers.

The work continued and resulted in an updated publication, "Journey to the Center of Software Supply Chain Attacks" [15] which identified 117 unique attack vectors, and 33 security controls related to mitigating the threats they represent.

One of the results of the work was the creation and publication of an attack tree organized in a hierarchical model[16]. At each hierarchical level, the identified attack vectors are listed, which are detailed in specific vectors at lower levels of the tree. The relevant security controls are associated with each vector at all levels.

The work details security control types (directive, predictive, detective, or corrective) and those responsible for their application (Package maintainer, Service provider, Package consumer).

The initial level of the attack tree has the following general cases: Develop and Announce New Malicious Packages, Create Name Confusion against Legitimate Packages, and Subvert Legitimate Packages.

*2) SLSA:* SLSA (Supply-chain Levels for Software Artifacts) (Open Source Security Foundation (OpenSSF), 2023) is a framework designed by Google for use in its internal projects, since 2013, and made available to the community. It aims to provide mechanisms for generating evidence that software supply chain protection steps have been carried out for a given project.

Although it does not explicitly define a taxonomy of attacks or threats, it lists the threats to the supply chain that it considers relevant (using the term "supply chain threats"), dividing them into threats to sources, dependencies, and the compilation environment. Defines a total of eight distinct threat subtypes.

Each of the main types of threat is represented through trails, and those proposed are distributed in increasing levels of quality in terms of security. Currently, only the build environment track has these controls defined, distributed between the L1, L2, and L3 levels. In cases where there are no controls implemented, the project is considered to be at SLSA level 0.

*3) S2C2F:* S2C2F (Secure Supply Chain Consumption Framework) is a software supply chain security framework that is based on a threat-based risk reduction approach, especially considering the consumption of dependencies on open source software projects, such as such as projects hosted on NuGet and NPM platforms.

It is structured around the following concepts:

- Control the ways of obtaining artifacts, such as cloning from source code repositories such as Git, downloading these artifacts directly from their original source using tools such as “wget”, and similar techniques. The framework recommends standardizing these methods across the organization so that different development teams use managed consumption flows.
- Continuous process improvement. It defines a maturity model, composed of four levels, thus allowing its incremental application in search of obtaining the desired level of security in the software supply chain.
- Scale. It provides tools that do not require a centralized control structure, in order to enable the organization to consume open-source software on a large scale.

Being a threat-based risk reduction framework, although it does not formally propose a taxonomy, it also defines a list of 13 threats to the security of the software supply chain. However, unlike the SLSA framework, it does not group such threats into types of threats related to the same topic, or to defined stages of the software supply chain.

*4) Enduring Security Framework (ESF):* The ESF is an initiative of North American security agencies, in partnership with the private market, organized by the National Security Agency (NSA), which aims to analyze risks to critical infrastructure and propose recommendations for the treatment of these risks [17].

In this context, it prepares and publishes recommendations on the security of critical infrastructure elements, among which we highlight Securing the Software Supply Chain for Customers, Securing the Software Supply Chain for Developers, and Securing the Software Supply Chain for Suppliers.

The first document is aimed at organizations in their software acquisition tasks, focusing on defining good practices for defining requirements for ordering software development or acquisition, implementation aspects of the selected software, such as product acceptance, functional testing, integration, team training, which is beyond the scope of this study.

The third document, although it addresses aspects related to software distribution, also addresses aspects related to software production in accordance with the specific requirements established by the customer, the treatment of vulnerabilities reported by customers relating to software made available by the distributor, and other issues that, likewise, are beyond the scope analyzed here.

The second document defines four main categories of threats to the software supply chain and proposes a series of security controls to mitigate the risks associated with them. The four categories are as follows:

a) An adversary intentionally injects malicious code into a product, or a developer does so unintentionally;
b) Incorporation of vulnerable third-party source code or binaries, intentionally or unintentionally;
c) Exploitation of weaknesses in the compilation process, in order to inject malicious software into a product component;
d) Modifying a product already in its distribution mechanism, resulting in the injection of malicious software into the original package or customer-deployed update package.

*5) Other frameworks and best practice guides:* Other frameworks and best practice guides focused on software supply chain security were also evaluated, but none of them propose the schematization of a taxonomy of attacks and threats, which is why they are not detailed here.

However, it is important to mention them, as they provide recommendations for relevant security controls applicable to the topic. Are they:

a) Cloud Native Computing Foundation (CNCF) - Software Supply Chain Best Practices - Document that addresses controls for protecting source code, dependencies, the compilation process, software artifacts, and the software delivery process [13];
b) OWASP SCVS (Software Component Verification Standard) - Framework that brings a group of controls, grouped into families, to be used to define, build, and verify the integrity of the software supply chain [18]

c) CIS Software supply chain security guide - It is a guide that provides security recommendations for the use of tools for building and deploying software through continuous integration and delivery (CI/CD) conveyors, in a DevOps environment [19].
d) OWASP top 10 CI/CD Security Risks - Similar to the CIS guide, it provides recommendations on security controls that should be implemented on the tools that make up the continuous integration and delivery (CI/CD) conveyors and focuses on the ten most relevant threats to this type of environment [20].

### *A. Other relevant initiatives*

In addition to the frameworks and best practice guides described here, other initiatives relevant to the challenge of tackling the issue of software supply chain security were identified, such as collections of Techniques, Tactics, and Procedures (TTPs) used in attacks on this environment.

Documenting TTPs is a step beyond threat mapping, as it details with a good degree of precision the procedures actually used by attackers in cases of real attacks. The evaluation of the mapped set of TTPs, in light of supply chain risk management, allows the organization to define a minimum set of TTPs of interest that must be subject to the implementation of security and monitoring controls.

a) Mitre Att&ck

Mitre Att&ck is not a framework specifically related to software supply chain security, but rather a broad knowledge base about TTPs, built from the observation of real attack cases (Mitre, 2023).

Although it does not explicitly define a taxonomy of attacks or threats to the software supply chain, it defines steps in the software or hardware supply chain where attacks can occur. These steps can be understood as a kind of threat taxonomy, and are as follows: Manipulation of development tools, Manipulation of a development environment, Manipulation of source code repositories (public or private), Manipulation of source code in open-source dependencies, Manipulation of software update/distribution mechanisms, Compromised/infected system images, Replacement of legitimate software with modified versions, Sales of modified/counterfeit products to legitimate distributors and Shipment interdiction.

Additionally, Mitre Att&ck provides examples of attack procedures used in real cases, and recommended security controls to prevent them.

b) Open Software Supply Chain Attack Reference

OSC&R, like Miter Att&ck, is also a compilation of TTPs, but specialized in cases of attacks on the software supply chain.

Defines a “Pipeline Bill of Materials (PBOM)”, evaluating possible attacks on all components of the software development and distribution pipeline. At the time of writing this article, the elements that make up the PBOM, as defined by OSC&R, are the following: Container Security, Open Source Security, SCM Posture, Secrets Hygiene, Code Security, Cloud Security, CI/CD Posture, Artifact Security, and Infrastructure as code.

Additionally, it groups the tactics, techniques, and procedures used for the attacks as follows: reconnaissance, resource development, initial access, execution, persistence, privilege escalation, and defense evasion.

## IV. Comparison of the approaches

This section compares and discusses the frameworks presented in the previous section. However, the initiatives in Section III-A are not covered since they are not focused on the threat landscape for the open-source software supply chain and protection approaches, but rather on the tactics, techniques, and procedures used to exploit the different vulnerabilities.

As mentioned in the descriptions of the frameworks in the previous topic, the works of Ladisa et al [15], the SLSA, and the ESF for developers categorize their taxonomies into levels and sublevels, seeking to better organize the sets of similar threats and the respective security controls they recommend.

However, their groupings are slightly different, so their comparison allows us to identify a more complete group of main threats, as illustrated in Table I.

In addition to these frameworks, there is also the S2C2F - Secure Supply Chain Consumption Framework, which lists a series of threats to the software supply chain but does not group them in a logical way like those compared here.

However, by analyzing each of the threats listed by the framework, it is possible to group them according to the same consolidated taxonomy detailed in Table I, as illustrated in Table II:

## V. Discussion of proposed security controls

The different frameworks and best practice guides cited, although based on similar taxonomies for defining the problem of security in the software supply chain, vary greatly in the depth of their approaches to applicable security controls, which can be observed, for example, from the quantities of recommended controls.

The SLSA and S2C2F frameworks have the least amount of controls.

| Ladisa et al [15] | SLSA | ESF for Developers | Consolidated Taxonomy |
|---|---|---|---|
| - x - | Source Threats | Adversary intentionally injecting malicious code or a developer unintentionally including vulnerable code within a product | Threats to source codes |
| Develop and Advertise Distinct Malicious Packages from Scratch. And create Name Confusion with Legitimate Package | Dependency Threats | Incorporating vulnerable third-party source code or binaries within a product either knowingly or unknowingly | Threats to dependencies |
| Subvert Legitimate Package | Build Threats | Exploiting weaknesses within the build process used to inject malicious software within a component of a product | Threats to the compilation process |
| - x - | - x - | Modifying a product within the delivery mechanism, resulting in the injection of malicious software within the original package, update, or upgrade bundle deployed by the customer | Threats to the distribution environment |

Table I: Comparison between taxonomies

SLSA currently defines nine controls relating only to the integrity of artifacts that belong to the build phase. However, this amount tends to increase as the detailing of controls over the other types of threats it foresees progresses (Threats to sources and dependencies).

S2C2F defines eight security controls relating to the consumption of secure artifacts throughout the software supply chain, that is, it has a greater focus on threats to dependencies, covering everything from the ingestion of selected artifacts into the organization's environment, through the other stages consumption,

| Consolidated Taxonomy | Threats according to S2C2F |
|---|---|
| Threats to source codes | The upstream source can be removed or taken down which can then break builds that depend on that OSS component or container |
| Threats to dependencies | Accidental vulnerabilities in OSS code or Containers that we inherit A malicious actor creates a malicious package that is similar in name to a popular OSS component to trick developers into downloading it. A malicious actor compromises a known good OSS component and adds malicious code into the repo Dependency confusion, package substitution attacks. An OSS component adds new dependencies that are malicious. Vulnerabilities/backdoors added to an OSS code base OSS components reach end-of-support/end-of-life and therefore don't patch vulnerabilities Vulnerability not fixed by upstream maintainer desired timeframe |
| Threats to the compilation process | A malicious actor compromises the compiler used by the OSS during build, adding backdoors |
| Threats to the distribution environment | The integrity of an OSS package is tampered with after build, but before consumption. A malicious actor compromises a distribution mirror of a package bad actor compromises a package manager account (e.g. npm) with no change to the corresponding open source repository and uploads a new malicious version of a package |

Table II: Threats related by S2C2F grouped according to the consolidated Taxonomy

up to the eventual correction of vulnerabilities found in these artifacts and the communication of this correction to the original maintainer.

Among the frameworks, OWASP SCVS is the one that proposes the greatest number of security controls. There are 87, distributed among six families of controls, with 3 incremental levels of verification of implementation of these controls, even enabling the certification of an organization by a third party in accordance with these levels. The six families of controls are as follows: Inventory, Software Bill of Materials (SBOM), Build environment, Package Management, Component Analysis, and Provenance and Pedigree.

On the other hand, the publications that classify themselves as Best Practice Guides delve more deeply into the recommendation and detailing of security controls.

The CIS Software Supply Chain Security Guide has 113 controls grouped into five distinct categories: source code, build pipelines, dependencies, artifacts, and deployment.

The OWASP Top 10 CI/CD Risks focuses on continuous integration and delivery (CI/CD) security, and recommends 57 security controls distributed among nine different themes that span from Flow Control Mechanisms through Identity Management and Access Controls, System Configuration, Artifact Integrity and Logging.

O ESF for Developers proposes 72 security controls for software supply chain security (excluding the policy definition stage and secure development management method), divided into the following categories: Develop Secure Code, Verify Third-Party Components, Harden the Build Environment, and Code Delivery.

Finally, the study by Ladisa et al [15], the only one that has as its specific objective the definition of a taxonomy of attacks and threats to the software supply chain, recommends 33 security controls, that would be capable of mitigating the 117 different threats mapped by it, its main focus being the issue of threats to dependencies.

## VI. Conclusions and Future Work

The issue of security in the software supply chain is a relatively recent concern for the industry, as evidenced by the publication dates of the frameworks and best practice guides evaluated in this study, which are distributed between the years 2021 and 2023. This results in that the approaches adopted by each one present variations, and that in some cases they are still incomplete works, as is the case with the SLSA framework.

However, it is possible to observe that there is some consensus between the main types of threats, which would refer to sources, dependencies, the buld environment, and software distribution.

The identified frameworks can be better used to help organizations structure their reactive software supply chain security efforts, while the best practice guides are more focused on recommending specific security configurations for the different steps involved in this supply chain.

As future work, we can list the analysis of best practices available for managing risks related to the software supply chain, especially as an input for the selection of frameworks to be adopted and security controls to be implemented by the organization, as well as a more detailed analysis of the costs and benefits of implementing the practices and controls proposed by the approaches analyzed, which could support the information security manager in prioritizing the implementation of controls that prove to be more efficient.

## References

[1] I. Lella, M. Theocharidou, E. Tsekmezoglou, A. Malatras, and S. García, *ENISA Threat Landscape for Supply Chain Attacks*. ENISA, 2021, [Accessed 29-09-2023]. [Online]. Available: https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks

[2] S. Peisert, B. Schneier, H. Okhravi, F. Massacci, T. Benzel, C. Landwehr, M. Mannan, J. Mirkovic, A. Prakash, and J. B. Michael, 3. [Online]. Available: https://ieeexplore.ieee.org/document/9382367

[3] J. Wetter and N. Ringland, "Understanding the impact of apache log4j vulnerability," 2021, [Accessed 29-09-2023]. [Online]. Available: https://security.googleblog.com/2021/12/understanding-impact-of-apache-log4j.html

[4] S. Moore, "Gartner top security and risk trends in 2022," 2022, [Accessed 29-09-2023]. [Online]. Available: https://www.gartner.com/en/articles/7-top-trends-in-cybersecurity-for-2022

[5] B. M. Beamon, "Supply chain design and analysis:: Models and methods," *International Journal of Production Economics*, vol. 55, no. 3, pp. 281–294, 1998, [Accessed 29-09-2023]. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925527398000796

[6] Open Source Security Foundation (OpenSSF), "Slsa • supply-chain levels for software artifacts," 2023, [Accessed 29-09-2023]. [Online]. Available: https://slsa.dev/

[7] J. Boyens, A. Smith, N. Bartol, K. Winkler, A. Holbrook, and M. Fallon, "Cybersecurity supply chain risk management practices for systems and organizations," *NIST*, 5 2022, [Accessed 29-09-2023]. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-161r1.pdf

[8] M. Souppaya, K. Scarfone, and D. Dodson, "Secure software development framework (SSDF) version 1.1," 2 2022, [Accessed 29-09-2023]. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-218.pdf

[9] "Building Security Maturity Model (BSIMM) Consulting Services | Synopsys — synopsys.com," 2023, [Accessed 29-09-2023]. [Online]. Available: https://www.synopsys.com/software-integrity/software-security-services/bsimm-maturity-model.html

[10] OWASP, "OWASP SAMM," 2022, [Accessed 29-09-2023]. [Online]. Available: https://owasp.org/www-project-samm/

[11] Microsoft, "Microsoft security development lifecycle," 2023, [Accessed 29-09-2023]. [Online]. Available: https://www.microsoft.com/en-us/securityengineering/sdl

[12] N. Ziv, L. Arzi, E. Paz, D. Cross, H. Suezawa, N. Penso, S. Sivan, D. Sahni, M. Kovalsky, C. Wang, R. Feintuch, H. H. Lavie, R. Atias, and G. Evron, "Open software supply chain attack reference (osc&r)," 2023, [Accessed 29-09-2023]. [Online]. Available: https://pbom.dev/

[13] Cloud Native Computing Foundation, "Software supply chain best practices," *CNCF*, 2021, [Accessed 29-09-2023]. [Online]. Available: https://project.linuxfoundation.org/hubfs/CNCF_SSCP_v1.pdf

[14] P. Ladisa, H. Plate, M. Martinez, and O. Barais, "Taxonomy of attacks on open-source software supply chains," *2023 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, US, 2023 pp. 1509-1526*, 2022, [Accessed 29-09-2023]. [Online]. Available: https://sap.github.io/risk-explorer-for-software-supply-chains/

[15] P. Ladisa, S. E. Ponta, A. Sabetta, M. Martinez, and O. Barais, "Journey to the center of software supply chain attacks," *IEEE Security & Privacy*, 2023, [Accessed 29-09-2023]. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10224821

[16] SAP SE, "Risk explorer for software supply chains," 2023, [Accessed 29-09-2023]. [Online]. Available: https://sap.github.io/risk-explorer-for-software-supply-chains/#/attacktree

[17] Enduring Security Framework, "Enduring security framework ESF," 2023, [Accessed 29-09-2023]. [Online]. Available: https://www.nsa.gov/About/Cybersecurity-Collaboration-Center/Enduring-Security-Framework/

[18] S. Springett, D. Russo, G. Fick, J. C. Herz, J. Scott, M. Symons, P. Nallapareddy, and B. Garcia, "Owasp software component verification standard v1.0," 2020, [Accessed 29-09-2023]. [Online]. Available: https://scvs.owasp.org/scvs/

[19] C. for Internet Security (CIS), "Software supply chain security guide," 2022, [Accessed 29-09-2023]. [Online]. Available: https://www.cisecurity.org/terms-of-use-for-non-member-cis-products/

[20] OWASP, "Owasp top 10 CI/CD security risks," 2022, [Accessed 29-09-2023]. [Online]. Available: https://github.com/OWASP/www-project-top-10-ci-cd-security-risks/blob/main/index.md