

# System & Software Security SBOM

Nuno Pereira  
João Pereira  
Davide Baggio

October 2024

We confirm that this report was fully produced by the team members **Davide Baggio, João Pereira and Nuno Pereira** and we are jointly responsible for all content presented in this work. All used sources were attributed properly.

Davide Baggio, João Pereira, Nuno Pereira

## Contents

1	Introduction	4
2	Overview of reviewed papers	5
3	Metrics defined	7
4	Analysis and Comparison of reviewed papers	8
5	Conclusion	9

# 1 Introduction

Software systems, now more than ever, outsource parts of application logic to first- or third-party *dependencies* — pieces of code that are used in conjunction or as a part of the application’s business logic but that are not part of the application itself. (Needs references)

Keeping track of dependencies can be an arduous task, which can be more easily managed using software tools and components such as *package managers* that keep track of an application’s dependencies and their versions (Needs references) , like *npm* or *cargo* [9, 10].

SBOMs [1], proposed by the U.S. National Telecommunications and Information Administration (NTIA), are a formal way of describing the software dependencies of an application and the relations between these dependencies. Other metadata can be attached to SBOM entries for a more comprehensive outline of the software components being depended upon.

Currently, there are 3 SBOM standards used in practice: OWASP CycloneDX [7], Software Product Data eXchange (SPDX) [8] and Software Identification Tagging (SWID) [6]. However, little consensus exists between these 3 standards, making for one of the many challenges against general SBOM adoption [2].

## 2 Overview of reviewed papers

To assess the state-of-the-art with respect to SBOMs and their industry-wide use, we conducted a critical review of some examples from the literature, which are enumerated and described below:

### **Software Bills of Materials Are Required. Are We There Yet?** [3]

In this paper, written by Zahan et al. and published in the *IEEE Security & Privacy (Volume: 21, Issue: 2, March-April 2023)*, the authors conducted a Grey Literature (GL) review of 200 internet articles, 100 for "challenges to adopt SBOM" and 100 for "benefits to adopt SBOM" in order to assess the biggest upsides and downsides experienced in practice regarding SBOM adoption. Grey literature was chosen as the majority of content regarding SBOMs in practice cannot be found in literature but rather in online articles and blog posts. The authors came up with 5 reported benefits from SBOM adoption and 5 challenges preventing SBOM adoption.

### **BOMs Away! Inside the Minds of Stakeholders: A Comprehensive Study of Bills of Materials for Software Systems** [5]

The paper, written by Stalnaker et al. and published in *ICSE '24: Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*, investigates the current state of Software Bills of Materials (SBOMs), which are recognized as vital tools especially after important incidents like the two mentioned ones: SolarWinds breach and Log4J vulnerabilities. The study identifies 12 major challenges related to SBOM creation and use, such as: insufficient tool support, SBOM maintenance difficulties and standard incompatibilities across different industries (as highlighted by the 138 interviews with stakeholders). The study identifies key SBOM standards (SPDX, CycloneDX, SWID) and emphasizes the need for better tools to facilitate SBOM creation, verification, and maintenance. Furthermore, it proposes 4 actionable solutions to overcome these critical problems and outlines future research directions aimed at maintaining SBOM accuracy over time and dealing with legacy systems.

### **An Empirical Study on Software Bill of Materials: Where We Stand and the Road Ahead** [2]

In this paper, written by Xia et al. and published in the *ICSE '23: Proceedings of the 45th International Conference on Software Engineering*, the authors aimed to understand the state of SBOM usage and adoption in practice, conducting 17 interviews and performing a survey based on the interviews. The authors gathered information about current SBOM practitioners and what these might feel is lacking in the industry regarding SBOM practices. The study is recent, provides a systematic methodology and provides perspectives from the Software Engineering standpoint.

### **On the Way to SBOMs: Investigating Design Issues and Solutions in Practice** [4]

This paper, written by Bi et al. and published in the *ACM*

*Transactions on Software Engineering and Methodology, Volume 33, Issue 6*, explores current practical uses and concerns/problems of SBOM in "the wild". The authors gathered data by mining several GitHub repositories and, out of those, discussions pertaining to the topic of SBOM. It was found that, generally, there are 4 phases to the SBOM lifecycle: planning, development, publication and maintenance.

### **3 Metrics defined**

In order to critically compare the reviewed papers, some metrics need to be used that can objectively rank these papers with respect to the metrics themselves.

Since SBOM is a relatively novel concept, comparing literature on this topic should focus on ...

As such, ...

## 4 Analysis and Comparison of reviewed papers



## 5 Conclusion

Supply Chain Security is an increasingly important factor of modern Software Development. Ensuring that companies adopt good practices in regards to the code they outsource is a pivotal step in ensuring the security requirements of software products. SBOMs (Needs references) are an example of such a practice that promotes good secure software engineering practices.

In this paper, we analyzed N examples of the state-of-the-art with regards to SBOMs and conducted a critical review of these papers according to our own metrics:

- Metric 1
- Metric 2
- ...

We found out that ...

Future work can expand on the methodology and results of this paper by ...

## References

- [1] Éamonn Ó Muirí. “Framing software component transparency: Establishing a common software bill of material (SBOM)”. In: *NTIA*, Nov 12 (2019). (accessed: 15.10.2024).
- [2] Boming Xia et al. “An Empirical Study on Software Bill of Materials: Where We Stand and the Road Ahead”. In: *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. 2023, pp. 2630–2642. DOI: 10.1109/ICSE48619.2023.00219.
- [3] Nusrat Zahan et al. “Software Bills of Materials Are Required. Are We There Yet?” In: *IEEE Security & Privacy* 21.2 (2023), pp. 82–88. DOI: 10.1109/MSEC.2023.3237100.
- [4] Tingting Bi et al. “On the Way to SBOMs: Investigating Design Issues and Solutions in Practice”. In: *ACM Trans. Softw. Eng. Methodol.* 33.6 (June 2024). ISSN: 1049-331X. DOI: 10.1145/3654442. URL: <https://doi.org/10.1145/3654442>.
- [5] Trevor Stalnaker et al. “BOMs Away! Inside the Minds of Stakeholders: A Comprehensive Study of Bills of Materials for Software Systems”. In: *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. ICSE ’24. Lisbon, Portugal: Association for Computing Machinery, 2024. ISBN: 9798400702174. DOI: 10.1145/3597503.3623347. URL: <https://doi.org/10.1145/3597503.3623347>.
- [6] National Institute of Standards and Technology (NIST). *Software Identification (SWID) Tagging — CSRC — CSRC — csrc.nist.gov*. <https://csrc.nist.gov/projects/Software-Identification-SWID>. [Accessed 15-10-2024]. (accessed: 15.10.2024).
- [7] Open Worldwide Application Security Project (OWASP). *OWASP CycloneDX Software Bill of Materials (SBOM) Standard — cyclonedx.org*. <https://cyclonedx.org/>. [Accessed 15-10-2024]. (accessed: 15.10.2024).
- [8] Linux Foundation. *SPDX Linux Foundation Projects Site — spdx.dev*. <https://spdx.dev/>. [Accessed 15-10-2024].
- [9] NodeJS Team. *NPM: Node Package Manager*. URL: <https://www.npmjs.com/>. (accessed: 15.10.2024).
- [10] Rust Team. *Cargo*. URL: <https://doc.rust-lang.org/cargo/>. (accessed: 15.10.2024).