System & Software Security
Software Bills of Materials

Davide Baggio - s4426428
João Pereira - s4443322
Nuno Pereira - s4443330

October 2024

Contents

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.

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.
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 [1], like *npm* or *cargo* [11, 12].
SBOMs (Software Bills of Materials) [2], 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 dependended upon.
Currently, there are 3 SBOM standards used in practice: OWASP CycloneDX [9], Software Product Data eXchange (SPDX) [10] and Software Identification Tagging (SWID) [8]. However, little consensus exists between these 3 standards, making for one of the many challenges against general SBOM adoption [3].
To ensure widespread practice of SBOM, potential challenges and their solutions must be known and categorized beforehand in order to mitigate issues during development that might arise from a lack of understanding of the software components that make up a product. However, since the topic of SBOMs is still relatively new, there is not much research made into the topic that provide for a standardized view of the common practices in place, and thus further work is required.

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? [4] 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 [7] 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 [3] 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 practicioners 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 [5] 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.

Charting the Path to SBOM Adoption: A Business Stakeholder-Centric Approach [6] The paper analyzes the slow adoption of SBOM in improving transparency and security within software supply chains. The research identifies four key stakeholder groups—system integrators, software vendors, B2B customers, and individual developers, and also examines how their roles affect SBOM adoption. Through interviews with 16 representatives from these groups, the authors analyze the incentives, concerns, and barriers related to SBOM. System integrators and software vendors are more likely to adopt SBOMs, driven by compliance requirements and the potential to improve their reputation and the quality of supplied software. On the other hand, B2B customers and individual developers show less interest because they struggle to see its immediate value and face challenges in resources and complexity. The study reveals that the main obstacles to adoption include a lack of expertise, concerns over the time and effort required to maintain SBOMs, vulnerability misclassification, and financial costs. The paper recommends targeted regulatory interventions and improvements in SBOM tools to align incentives across stakeholders. The research concludes that while SBOMs have significant potential to enhance software security, widespread adoption will require external pressures and better tools to support stakeholders.

3 Metrics defined

To effectively compare the reviewed papers, objective metrics should be used to rank them based on those specific criteria.
Since SBOM is a relatively novel concept, comparing literature on this topic should focus on finding out and exposing the most common challenges pertaining to SBOM widespread adoption and how those can be mitigated. Other interesting aspects relevant for SBOM adoption is how generalized these problems are, since problems that are too specific to a certain domain only provide value for that domain and cannot be reused in other contexts. As such, the metrics we have devised for the critical comparison of the papers we reviewed are:

Metrics of Quality

1. **Aims**: are the aims/research questions of the paper clearly defined?
2. **Conclusions**: are the conclusions drawn from the study findings valid and do they align with the aims of the paper?

Metrics of Quantity

1. **Number of SBOM adoption challenges**: the number of challenges an organization or system integrator needs to face before extracting value out of the use of SBOMs directly correlates to how eager they might be to adopting SBOMs in their processes: if the benefits don't outweigh the challenges, it is not valuable, and thus not desirable, to put in the effort required to correctly adopt and practice SBOM development.
2. **Frequency of challenges**: the number of times common challenges appear across different research papers, either quantitatively or qualitatively.
3. **Number of solutions to standardize SBOM use**: the number of solutions proposed by the paper to standardize SBOM use is a good indicator of how much thought the authors have put into the topic and how much they have researched the topic.

Metrics of methodological soundness

1. **Sampling**: is the sampling of participant clearly defined and does it provide a representative sample of the population the paper is studying?
2. **Analysis of study findings**: are the analytic methods clear, systematic and reproducible?

Each of the defined metrics will be rated on a scale from 1 to 5, where 1 means the metric is not met at all, and 5 means the metric is met perfectly.

3.1 Considerations on the defined metrics

Although we believe these metrics to be valid comparison points between the papers discussed, there are some potential issues which could hinder the validity of the comparisons made with them:

- For the **Number of SBOM adoption challenges** metric, one shortcoming of simply counting the challenges presented by each paper is that different authors might group their findings in different ways, so what would be, for example, 1 challenge for one author could become 2 for a different author. The same reasoning applies to the **Number of solutions to standardize SBOM use** metric. We did our best to analyze the papers with as much scrutiny regarding these issues as possible.

4 Analysis and Comparison of reviewed papers

Having defined a set of relevant metrics for the topic of SBOMs, we analyze the several papers according to them, using the results to compare them.
Our findings can me summarized as follows:

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 [2] 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] Diomidis Spinellis. "Package Management Systems". In: *IEEE Software* 29.2 (2012). [Accessed 21-10-2024], pp. 84–86. DOI: 10.1109/MS.2012.38.

[2] Éamonn Ó Muirí. "Framing software component transparency: Establishing a common software bill of material (SBOM)". In: *NTIA, Nov* 12 (2019). [Accessed 15-10-2024].

[3] 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)*. [Accessed 15-10-2024]. 2023, pp. 2630–2642. DOI: 10.1109/ICSE48619.2023.00219.

[4] Nusrat Zahan et al. "Software Bills of Materials Are Required. Are We There Yet?" In: *IEEE Security & Privacy* 21.2 (2023). [Accessed 15-10-2024], pp. 82–88. DOI: 10.1109/MSEC.2023.3237100.

[5] 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). [Accessed 15-10-2024]. ISSN: 1049-331X. DOI: 10.1145/3654442. URL: https://doi.org/10.1145/3654442.

[6] Berend Kloeg et al. "Charting the Path to SBOM Adoption: A Business Stakeholder-Centric Approach". In: *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*. ASIA CCS '24. [Accessed 15-10-2024]. Singapore, Singapore: Association for Computing Machinery, 2024, pp. 1770–1783. ISBN: 9798400704826. DOI: 10.1145/3634737.3637659. URL: https://doi.org/10.1145/3634737.3637659.

[7] 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. [Accessed 15-10-2024]. Lisbon, Portugal: Association for Computing Machinery, 2024. ISBN: 9798400702174. DOI: 10.1145/3597503.3623347. URL: https://doi.org/10.1145/3597503.3623347.

[8] 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].

[9] Open Worldwide Application Security Project (OWASP). *OWASP CycloneDX Software Bill of Materials (SBOM) Standard — cyclonedx.org*. https://cyclonedx.org/. [Accessed 15-10-2024].

[10] Linux Foundation. *SPDX Linux Foundation Projects Site — spdx.dev*. https://spdx.dev/. [Accessed 15-10-2024].

[11] NodeJS Team. *NPM: Node Package Manager*. [Accessed 15-10-2024]. URL: https://www.npmjs.com/.

[12] Rust Team. *Cargo*. [Accessed 15-10-2024]. URL: https://doc.rust-lang.org/cargo/.