Considerations for Managing Challenges in Software Bill of Materials (SBOM) Data Normalization

April 2026

MITRE

Notice

This white paper was prepared by The MITRE Corporation under contract with the U.S. Food and Drug Administration (FDA). The views, opinions, and findings contained in this white paper do not constitute agency guidance, policy, recommendations, or legally enforceable requirements. Utilizing the information presented in this document does not constitute compliance with any requirements of the Federal Food, Drug, and Cosmetic Act, or any other applicable law. Identification of commercial websites, products, or services in this paper does not constitute an endorsement by FDA.

For further information, please contact The MITRE Corporation, Contracts Management Office, 7515 Colshire Drive, McLean, VA 22102-7539, (703) 983-6000.

Authors

Melissa P. Chase
Steven Christey Coley
Margie Zuk

The MITRE Corporation, Bedford, MA

Acknowledgments

MITRE would like to thank the medical device manufacturers, SBOM tool developers and service providers, cybersecurity and regulatory consultants, and private-sector partnerships that provided insights into SBOM data normalization challenges and their approaches for managing them. The considerations and resources provided in this document are a direct result of lessons learned from these engagements.

Executive Summary

In October 2024, MITRE published *Data Normalization Challenges and Mitigations in Software Bill of Materials (SBOM) Processing: A White Paper for Medical Device Manufacturers,* which highlighted the data normalization challenges arising when processing and managing SBOMs at scale, and offered some high-level mitigation strategies. This paper builds upon the earlier one and provides more detailed considerations on implementing approaches to address data normalization challenges, focusing on putting technologies and processes in place that can evolve with developing SBOM technologies and changes in organizational structures.

Section 1 provides background information.

Section 2 describes our approach, including conducting a targeted landscape analysis.

Section 3 focuses on the evolution of SBOM tools, including:

- The evolution of SBOM tool capabilities for generation, consumption, analysis, and management.
- How data normalization contributes to inconsistencies across tools.
- Considerations when acquiring tools.
- Section 4 discusses managing a “source of truth” (SoT) to provide a consistent nomenclature throughout an organization, including:
- Processes for creating and managing an organization’s SoT, building upon the different tooling approaches discussed in Section 3.
- Sources and considerations in creating a SoT for the different SBOM baseline attributes.

Table of Contents

1 Introduction

1.1 Background

MITRE published Data Normalization Challenges and Mitigations in Software Bill of Materials (SBOM) Processing: A White Paper for Medical Device Manufacturers in October 2024 (hereinafter referred to as “SBOM Data Normalization Challenges white paper”) [1]. One of the key challenges that was identified was the continuing evolution of SBOM standards and tools, which could lead to Medical Device Manufacturers (MDMs) evolving their processes for generating and managing SBOMs and migrating to new tooling. The paper suggested that MDMs maintain a central database with a canonical nomenclature to facilitate matching attributes across multiple SBOMs (received or generated) and external repositories, such as vulnerability databases, to minimize additional normalization challenges created by tool migration.

1.2 Purpose

This paper will provide additional details on addressing data normalization challenges when generating SBOMs. Since MDMs are at different levels of maturity in SBOM management and have different numbers of product lines, the paper will discuss a range of options used in managing and generating SBOMs, including manual approaches (with the aid of spreadsheets and ad hoc scripts[1]), open-source tooling, and commercial SBOM management tools.

2 Approach

In developing this paper, MITRE conducted a landscape analysis; participated in the Cybersecurity and Infrastructure Security Agency (CISA) SBOM Community of Interest and reviewed its interim and final products; and interviewed a broad range of stakeholders, including MDMs, cybersecurity and regulatory consultants, participants in SBOM standardization efforts, and SBOM tool vendors.

A specific recommendation from the 2024 SBOM Data Normalization Challenges white paper [1], was to maintain a central database to help resolve normalization issues. The interviews MITRE has since conducted indicate that SBOM tools have been maturing rapidly and many tools go beyond source code analysis to create an SBOM and provide support for resolving data normalization issues and managing SBOMs. As a result, in this work MITRE focuses on how tools could be leveraged, either using available commercial or open-source tools, or adapting practices found in tooling to in-house approaches, and how to create the canonical nomenclature to support data normalization.

[1] Spreadsheets and scripts may used by MDMs to analyze the software and identify the baseline attributes for the SBOM. It should be noted that FDA recommends machine-readable formats for SBOMs that are submitted with premarket submissions.

In this paper we will:

- Describe the state of SBOM tools, including resources for finding appropriate tools and recommendations for evaluating tools (e.g., questions to ask vendors, using the challenges described in the SBOM Data Normalization Challenges white paper) to assess handling of normalization issues.
- Discuss SBOM management considerations, including creating an internal capability (e.g., parsers, databases) or using a commercial tool. Section 4 includes general considerations in managing SBOM data and issues for specific attributes.
- Identify internal and external sources for developing canonical representations for various SBOM attributes, including supplier, component name, and software identifiers.

3 SBOM Tooling and Data Normalization

As adoption of SBOM increases globally, including growing requirements, whether through the MDMs internal process or external requirements, the SBOM tool landscape continues to evolve and may include:

- Manual methods performed in cases for which there is limited tooling available (e.g., embedded systems or C/C++ code for which there are no common packaging and distribution frameworks).
- MDM-developed custom mechanisms, such as scripts or spreadsheets intended to help automate or partially automate SBOM production.
- Use of open-source tools, potentially with custom “glue code” to link multiple tools together.
- Use of commercial tools.
- Tools to integrate with the organization’s own asset-management infrastructure (e.g. a Configuration Management Database (CMDB)).

MDMs may work closely with their tool vendors to develop solutions and build up their knowledge to help effectively maintain a Central Alias Database as advocated in our SBOM Data Normalization Challenges white paper [1]. The data and techniques, as implemented by the vendor, can then be fed back into the SBOM tool in a way that benefits all customers.

Below we describe the evolution of SBOM tool capabilities for generation, consumption, analysis, and management (Section 3.1), discuss the inconsistencies in the SBOMs generated by different tools (Sections 3.2 and 3.3), and include some considerations to account for when acquiring SBOM tools (Section 3.4).

3.1 Evolution of Tool Types / Adoption

SBOM-related tools are rapidly evolving and offer a variety of capabilities [2]. Tools can:

- Produce SBOMs – Through close integration into the build or deployment process, analysis of software artifacts such as source code, or manually editing SBOMs.
- Ingest SBOMs – Through capabilities such as imports, comparison between SBOMs, and view/display.
- Transform and Merge SBOMs – By translating them to different formats, merging multiple SBOMs, or providing support to other tools.

Note that for most types of tools, there are both open-source and commercial options available.

Many SBOM tools have multiple capabilities or are being used in different locations within the software supply chain [3]. This can make it difficult and time-consuming for MDMs to identify their key requirements and select the appropriate tools for their needs.

As SBOM tools have matured and the "SBOM management" market has become more competitive, there has also been growing inconsistency in names and definitions for important data elements within SBOMs, which introduces normalization challenges. However, community-based efforts such as the Tooling and Implementation Working Group hosted by CISA [4] have sought to reduce this inconsistency.

As the need to manage SBOMs on a large scale has grown (e.g., to handle multiple product lines as an MDM, or to track assets as a healthcare delivery organization), "SBOM management" tools have begun to emerge. These tools typically provide capabilities that are similar to the Central Alias Database as referenced in [1], such as alias tables, alias rules, matching algorithms, or other metadata. SBOM management tools face normalization issues because they collect and manage SBOMs that were generated from many different sources and tools. These SBOM management tools typically have built-in capabilities that address normalization issues directly or provide mechanisms to the user to manually resolve any uncertainties.

MDMs often depend on SBOM management tools to handle such normalization issues for them, but they will still perform manual analysis or review, and there may be some gaps for some components due to lack of availability of tooling to produce SBOM (e.g., for embedded software). To address these issues, some MDMs work closely with the tool vendors to address various problems, including normalization.

The open-source community is actively developing open-source tools that can generate SBOMs as part of the build process and encouraging SBOM use in open-source projects. Projects such as the Open Source Security Foundation's (OpenSSF) SBOMs Everywhere project provide catalogs of available open-source tools as categorized by their capabilities, support for standards such as SPDX and CycloneDX, etc. However, normalization capabilities are not usually identified as a desired feature.

Some MDMs have not adopted management tools offered by vendors and may perform large-scale SBOM management themselves by having a centralized service that supports independent product teams across the MDM.

3.2 SBOM Repositories for Analysis and Comparison

With the growth of SBOM tool development and use, various parties have begun collecting, collating, and comparing SBOMs, although the focus has been more on inconsistencies in tool results without any explicit emphasis on normalization. In 2024, the Carnegie Mellon University Software Engineering Institute (SEI) organized an SBOM Harmonization Plugfest involving multiple participants who produced SBOMs for a fixed set of products [5]. A cursory analysis of the Plugfest's resulting data repositories[2] shows that participants often produced significantly different SBOMs for the same products being analyzed. Arguably, the major inconsistencies, such as variance in numbers of components, handling dependencies, and SBOM build processes, suggest more substantive problems to address before tackling normalization, although SEI observes that "lack of normalization" contributes to the variance [6].

Separately, there have been efforts to collect and/or generate large-scale SBOM repositories, such as the dataset created by Cahora and Camp [7], which collected SBOMs from over 100,000 GitHub repositories as "a foundational resource for advancing empirical research in software supply chain security and transparency," or Kishimoto, et al.'s dataset [8], which generated SBOMs for Java products "to evaluate tools that utilize SBOMs [since] research on SBOM consumption tools remains limited." Because such repositories are relatively new, it is not clear how they could best be used to understand SBOM tool capabilities, and whether such repositories could be used to identify normalization issues.

3.3 Normalization-Specific Issues for SBOM Management

Based on the interviews we conducted, it appears that:

- Few MDMs indicate that they are facing normalization issues. This may reflect relative immaturity of the industry, and MDMs who have just started to produce SBOMs may not have had to face this problem yet. Alternately, this may reflect a dependence on a single SBOM tool, which may inherently avoid many normalization issues because they are effectively using or producing consistent data.
- Representatives of some large, mature third-party vendors have acknowledged that they face normalization issues on a regular basis, as well as some SBOM tool vendors. These are suppliers to MDMs, and not the MDMs themselves.
- Larger MDMs with multiple products and separate teams for each individual product line generally generate and maintain their own SBOMs. Normalization issues may arise when managing the MDM's entire repository, but this was not emphasized in interviews. For some MDMs with few product lines or MDMs that are still developing their first product, normalization issues were typically handled manually and did not seem to be considered a significant burden. Because smaller or newer MDMs did not have as much legacy code and, presumably, smaller code bases, it could be that integration of SBOM information from multiple sources or components was less demanding.

One capability that has been important for some SBOM tool vendors is the ability to recognize and parse software identifiers, versions, etc. Custom parsers and custom logic are sometimes used. For example, while semantic versioning is common, there are many versioning schemes that do not follow strict syntactic or temporal patterns, requiring some custom knowledge or logic to address, such as

[2] https://github.com/cmu-sei/sbom-plugfest-2024

for cases in which version matching is needed for vulnerability management. Some kinds of software can be difficult to differentiate from bundled or packaged products, such as containers and Yocto distributions. In other cases, the software identifier schemes might not cover the components to the desired granularity, such as Android components.

To generate, ingest, aggregate, and manage SBOMs at scale, one consideration is that parsing should be as automated as possible, although there is a recognition that manual, human intervention is often needed, and there is occasionally some reluctance to trust tools too much. The desire for automation has led to experimentation (and sometimes adoption) of Artificial Intelligence / Machine Learning (AI/ML) technologies to support SBOM processing and generation. However, there is a general acknowledgment that such technologies are subject to both false positives and false negatives, and manual review is necessary. We note that in the general vulnerability management industry, there is increased experimentation and interest in AI/ML to extract key information about vulnerabilities due to issues of scale in which human analysis is not necessarily able to keep up with the volume and speed [9] [10].

3.4 Considerations for Tool Suitability

Our interviews did not concentrate on how MDMs evaluated SBOM management tools to determine which tools were most suitable for their use with respect to normalization. However, based on findings as covered in Sections 3.1 and 3.3, the following considerations could be made when developing or acquiring tools for handling SBOMs.

- How does the tool handle version numbers, especially unusual version schemes? Some SBOM tool vendors needed to create a wide variety of version-matching algorithms for separate components. Any inconsistencies in these algorithms between vendors could introduce normalization problems.
- Does the tool provide a label or other indicator when it has made an automated decision that may be subject to false positives? The lack of a label may lead to excessive dependence on incorrect data.
- How does the tool involve the human user in decision making for choosing how to normalize a previously unseen, new data element, such as a supplier name?
- Is there a mechanism for sharing data across different users or organizations? To reduce the amount of manual intervention, knowledge bases could be fed from multiple organizations, so that future SBOM processing would integrate information from previous efforts. Note that MDMs might wish to limit use of such data, so confidentiality considerations would be important. Without mechanisms for sharing data, separate users or organizations could have inconsistent results.
- Can collected data be shared and, if necessary, migrated to other tools that may be acquired in the future? Since the industry is growing rapidly, it is likely that new tools may arise that would better meet an organization's needs.
- Can the tool's data stores (e.g., alias database, etc.) be easily browsed and audited?
- What is the tool's behavior when facing multiple conflicting or complementary identifiers (e.g., both a CPE and PURL are known)? Different users may have different preferences, leading to inconsistencies across multiple SBOMs.

- For capabilities in which raw data is automatically analyzed and transformed (e.g., in normalization of supplier names and/or comparing product versions), is the logic auditable and understandable? This question may be most important in the case of AI/ML, in which explanation of conclusions may be difficult.
- How closely can the organization work with the SBOM tool vendor? Some MDMs work closely with their vendors, who address their concerns. This may be an important consideration while the industry is still in its early stages, and while MDMs do not have full clarity about their own product requirements.

4 SBOM Management and Data Normalization

As mentioned in Section 3.1, one of the key recommendations in the SBOM Data Normalization Challenges white paper [1] was to maintain a central "source of truth" (SoT) to support creating the SBOM baseline attributes that can be used to identify components and their relationships, especially Supplier Name, Component Name, Version String, and Unique Identifier. This SoT could help mitigate the challenges created by the rapidly developing SBOM tool space as an organization evolves their processes to adopt new tools that better fit their needs. It can also mitigate challenges in generating and managing SBOMs created by changes in technology stacks and development environments, and absorbing products through mergers and acquisitions.

The SBOM Data Normalization Challenges white paper [1] focused on developing an aliases database to provide a single SoT, but there are multiple approaches, including parsing, fuzzy matching, and databases, to provide the desired functionality:

- Retrieve the canonical name of a component, supplier, etc. (e.g., what are the canonical names of all operating systems used in an MDM's products?).
- Given the canonical name of an entity, provide other names that refer to the entity.
- Given a name, retrieve its canonical name.

This common nomenclature capability could be managed by individual business units or by the enterprise. Centralization provides efficiency: once a product development team resolves nomenclature issues and matching issues (e.g., false positives and false negatives), that information can be available to other teams. For large MDMs, who may reuse software components across business units, enterprise centralization could be a force multiplier.

4.1 Considerations for Creating a "Source of Truth" (SoT)

This sub-section discusses general considerations in creating a SoT.

Define use cases. Identify the use cases that the SoT will support, such as:

- Aggregating SBOMs across business units, including from mergers and acquisitions.
- Validating SBOMs from third-party suppliers.
- Incorporating SBOMs from open-source software.
- Searching vulnerability databases and alerts for vulnerability management.

These use cases will help define the requirements for the SoT, such as:

- The services that will be provided.
- The components of an application programming interface (API).
- The elements used for different functionality (e.g., what elements are used to match the software component: software identifiers, combinations of baseline attributes, and additional metadata).

Determine technical approach. As discussed in Section 3.3, there is a broad range of approaches to managing SBOMs. An MDM could pick an approach that is appropriate for their size, number of products, organic skill set, and other factors. The approach could be based entirely on internal development (perhaps leveraging open-source tools), a commercial tool, or some combination. The SBOM Data Normalization Challenges white paper [1] described many data normalization challenges, general and attribute-specific, which can help guide tool selection, tool/script development, and test case development.

If the chosen technical approach involves internal development, the MDM might consider leveraging open-source components, both SBOM-specific tools and more general tools (e.g., instead of building a parser from scratch, the MDM can find an open-source library most appropriate for the organization's development environment).

If the chosen technical approach is to procure a third-party tool, the MDM can use the questions in Section 3.4 to guide the selection process. Since no tool works for all cases, manual resolution may likely be performed when matching fails, so it is important to understand how this process will work, what support will be provided by the tool vendor, etc. The following catalogues of SBOM tools are available:

- OpenSSF's SBOM Everywhere Working Group's.[3]
- Tooling supporting CycloneDX.[4]
- Tooling supporting SPDX.[5]

[3] https://sbom-catalog.openssf.org/catalog/#/Circle/Normalize/HowTo

[4] https://cyclonedx.org/tool-center/

[5] https://spdx.dev/use/spdx-tools/

Define schemas and API. It is recommended to define an API for the services that are provided by the SoT. By having a well-defined API, the underlying implementation can change in the future, and the users of those services will still be able to access the services. If there is an internally maintained database, MDMs can create schemas, which define the entities and relationships of the database. The schema and API elements are derived from the use cases. In defining the API, MDMs should consider approaches to reduce friction with the received and generated SBOMs. For example, an MDM can choose to standardize on a single SBOM format (e.g., SPDX or CycloneDX) and serialization (e.g., JSON or XML), and use that in the API, providing mechanisms to convert inputs in other formats to the desired format and serialization if necessary.

Define processes for updating and ensuring consistency. The information in the SoT is likely to be updated to reflect the resolution of issues when matching or correlating raw SBOM information against the SoT, so in the future the matching/correlation process will work automatically. In addition, the SoT will likely be updated to reflect external changes, such as changes in supplier name, component name, or versioning scheme. Metadata can be included to determine the time periods when the different attribute values are valid.

In addition, it is important to keep different elements synchronized within the SoT. For example, software identifiers may include supplier names, component names, and versions, and if the SoT is maintaining the canonical versions of those attributes, the MDM can define processes to ensure these elements are aligned.

4.2 Considerations for Key Baseline Attributes

The key baseline attributes for identifying software are supplier name, component name, version, and software identifiers. As mentioned above, the SBOM Data Normalization Challenges white paper [1] describes the data normalization challenges that may be encountered when matching and correlating raw SBOM data against the normalized information in the SoT. As mentioned in Section 4.1, these challenges can lead an organization to develop an approach to establishing a consistent nomenclature, by identifying issues to consider (e.g., case, separators, abbreviations).

The third edition of the SBOM Framing Document [4] may be useful in defining the canonical nomenclature for the SoT. Although the Food and Drug Administration (FDA) guidance document “Cybersecurity in Medical Devices: Quality System Considerations and Content of Premarket Submissions” (hereinafter referred to as FDA premarket cybersecurity guidance) [11] recommends that MDMs provide machine-readable SBOMs consistent with the baseline attributes identified in the second edition of the SBOM Framing Document [12], the descriptions of the attributes in the second edition were sometimes incomplete. The third edition further defines and clarifies the baseline attributes offering fuller descriptions of the minimum information to include, recommended practices, and aspirational goal for each attribute, which MDMs can use in defining their own canonical nomenclature.

Supplier Name. The Supplier Name is the name of the entity that supplies a component. If the component is unmodified, the Supplier Name is the legal entity name for commercial software or the project name for open-source software. See [4] for additional nuances. Sources for Supplier Name include:

- Internal corporate contracting databases.
- Licenses and copyrights embedded in the software component.
- External databases:
  - National Vulnerability Database (NVD).
  - Securities and Exchange Commission Filings.
  - Database of state corporation registry databases.

Component Name. The Component Name is the public name of a component, as provided by the component's supplier. Sources for Component Name include:

- Internal corporate contracting databases.
- Licenses and copyrights embedded in the software component.
- External databases of software identifiers, such as Common Platform Enumeration (CPE) and Package Uniform Resource Locator (PURL) since the Component Name is part of some identifiers (see below).

Version String. The Version String identifies the version of a component. The SoT could include component versions, for example, if one of the services it provides is to determine if a medical device contains the version of a component affected by a vulnerability. If the SoT doesn't include specific versions, it could contain metadata about the versioning scheme (e.g., what type of versioning scheme is used, whether version information is recorded as individual versions or version ranges, and when version schemas have changed).

Unique Identifier. The Unique Identifier provides additional information to help uniquely define a component. The two most widely used identifiers are CPEs and PURLs. Most of the stakeholders we interviewed prefer using CPEs when available, since it is the identifier used by NVD, and so facilitates vulnerability management. Since CPEs are typically only generated when a vulnerability is discovered, a component might not have a CPE. In that case, the SBOM producer might fall back to a PURL, if available. If the component is a package available through a package manager, a PURL exists with the package type associated with the package manager/ecosystem, and if the component is software available in a bare repository, a "github" or "generic" PURL can be created.[9] Finally, the SBOM producer might create a "pseudo-CPE," marked as non-authoritative, or another proprietary format. In the future, SBOM producers might be able to use the proposed PURL Software Component Identification type,

[6] https://nvd.nist.gov/

[7] https://www.sec.gov/search-filings

[8] https://www.northwestregisteredagent.com/

[9] See the definition of PURL Type definitions in the PURL specification (https://github.com/package-url/purl-spec/blob/main/PURL-TYPES.rst).

which is intended to be used for commercial software, standalone open-source projects, internally developed software, or binary-only code.[10]

Authoritative databases for CPEs and PURLs are:

- NVD's CPE Dictionary.[11]
- AboutCode's PURL Database.[12]

CPEs can be used to query NVD for vulnerability information about a software component within an SBOM. If a component only has a PURL as its unique identifier, an MDM might want to translate PURLs to CPEs, especially when automating vulnerability management processes. There are two open-source projects that map PURLs to CPEs, that could help automate queries against NVD for vulnerability information about components:

- AboutCode's PURL Database has the ability to generate a mapping of PURLs to CPEs and has published a repository for CVEs from 1999–2022.[13]
- ScanOSS has released their purl2cpe tool as open source, along with their current database.[14]

5 Summary

SBOMs are a powerful tool in software security and software supply chain risk management of medical devices. In our SBOM Data Normalization Challenges white paper [1], we described the normalization issues that hinder the effectiveness of the generation and consumption of SBOMs and suggested that SBOM data, especially the baseline attributes and additional data recommended in the FDA premarket cybersecurity guidance [11], are normalized using a consistent nomenclature and data formats. In this paper we provide additional considerations on how to create a consistent nomenclature, and correlate and aggregate SBOMs, when generating SBOMs and using them to support vulnerability management. This paper focuses on putting technologies and processes in place that can evolve with developing SBOM technologies and changes in organizational structures.

SBOM tools and management processes are rapidly evolving, and the information in this paper is intended to help medical device manufacturers evolve their tools and processes to take advantage of newer SBOM technologies, and also to address the challenges to SBOM generation created by evolving development technologies, introducing new products, and managing mergers and acquisitions.

[10] https://github.com/package-url/purl-spec/issues/516 is the GitHub issue describing the new type, which will be part of the PURL specification submission to the European Computer Manufacturers Association (ECMA).

[11] https://nvd.nist.gov/products/cpe

[12] The VulnerableCode database (https://public.vulnerablecode.io/) aggregates package information from the authoritative package mangers and has a package search feature. It's also possible to stand up an internal PURL database using the code from AboutCode's purldb (https://github.com/aboutcode-org/purldb).

[13] See footnote [12] for the PURL database. https://github.com/aboutcode-org/vulnerablecode-purl2cpe is the GitHub repository of the PURL to CPE mappings, along with instructions on generating the data.

[14] https://github.com/scanoss/purl2cpe

Glossary

Acronym	Definition
AI	Artificial Intelligence
API	Application Programming Interface
CISA	Cybersecurity and Infrastructure Security Agency
CPE	Common Platform Enumeration
CVE	Common Vulnerabilities and Exposures
CycloneDX	Cyclone Data Exchange
ECMA	European Computer Manufacturers Association
FDA	Food and Drug Administration
FFRDC	Federally Funded Research and Development Center
JSON	JavaScript Object Notation
MDM	Medical Device Manufacturer
ML	Machine Learning
NVD	National Vulnerability Database
OpenSSF	Open Source Security Foundation
PURL	Package URL
SBOM	Software Bill of Materials
SoT	Source of Truth
SPDX	Software Package Data Exchange
URL	Uniform Resource Locator
XML	eXtensible Markup Language

References

[1] [The MITRE Corporation, “Data Normalization Challenges and Mitigations in Software Bill of Materials (SBOM) Processing: A White Paper for Medical Device Manufacturers,” 24 October 2024. [Online]. Available: https://www.mitre.org/news-insights/publication/data-normalization-challenges-mitigations-software-bill-materials-processing.

[2] NTIA SBOM Formats and Tooling Working Group, "SBOM Tool Classification Taxonomy," 30 March 2021. [Online]. Available: https://www.ntia.gov/files/ntia/publications/ntia_sbom_tooling_taxonomy-2021mar30.pdf.

[3] Community-led working group on SBOM Tooling and Implementation, "Types of Software Bill of Material (SBOM) Documents," April 2023. [Online].

[4] Tooling and Implementation Working Group hosted by CISA, "Framing Software Component Transparency: Establishing a Common Software Bill of Materials (SBOM), 3rd edition," 3 September 2024. [Online]. Available: https://www.cisa.gov/sites/default/files/2024-10/SBOM%20Framing%20Software%20Component%20Transparency%202024.pdf.

[5] Carnegie Mellon University Software Engineering Institute, "SBOM Harmonization Plugfest 2024," 2024. [Online]. Available: https://resources.sei.cmu.edu/news-events/events/sbom/.

[6] Carnegie Mellon University Software Engineering Insitute, "Software Bill of Materials (SBOM) Hamonization Plugfest (2024) Report," July 2025. [Online]. Available: https://www.sei.cmu.edu/library/software-bill-of-materials-sbom-harmonization-plugfest-2024/.

[7] A. Cahora and L. J. Camp, "Compilation of SBOM Dataset from 100 000+ Public GitHub Repositories," 4 May 2025. [Online]. Available: https://zenodo.org/records/15334733.

[8] R. Kishimoto, T. Kanada, Y. Manabe, K. Inoue, S. Qiu and Y. Higo, "A Dataset of Software Bill of Materials for Evaluating SBOM Consumption Tools," 9 April 2025. [Online]. Available: https://arxiv.org/abs/2504.06880.

[9] W. Hu and V. L. L. Thing, "CPE-Identifier: Automated CPE identification and CVE summaries annotation with Deep Learning and NLP," in International Conference on Information Systems Security and Privacy, 2024.

[10] I. Gutierrez and D. Crawford, "Using machine learning to improve software identification: Modern solutions to legacy challenges," 9 October 2024. [Online]. Available: https://www.cgi.com/sites/default/files/2024-10/swam_softwareidentification_whitepaper_noai_final_0.pdf.

[11] Food and Drug Administration Center for Devices and Radiological Health, "Cybersecurity in Medical Devices: Quality System Considerations and Content of Premarket Submissions," 27 June 2025. [Online]. Available: https://www.fda.gov/media/119933/download?attachment.

[12] NTIA Multistakeholder Process on Software Component Transparency Framing Working Group, "Framing Software Component Transparency: Establishing a Common Software Bill of Materials (SBOM), 2nd edition," 21 October 2021. [Online]. Available: https://www.ntia.gov/sites/default/files/publications/ntia_sbom_framing_2nd_edition_20211021_0.pdf.