

SBOM Document Quality Guide

Compliance Management Guide for the Supply Chain

An official guide published by the OpenChain Project (www.openchainproject.org)

Version 1.0

This page intentionally left blank

## Table of Contents

[Table of Contents 3](#_Toc212042308)

[Introduction 7](#_Toc212042309)

[0. Preface 9](#_Toc212042310)

[1. Scope and SBOM Document Quality 9](#_Toc212042311)

[2. Terms and Definitions 10](#_Toc212042312)

[3. Guidelines to Enhance SBOM Document Quality 10](#_Toc212042313)

[3.1. Data Format 10](#_Toc212042314)

[**3.1.1 Verification and reference material** 10](#_Toc212042315)

[**3.1.2 Rationale** 10](#_Toc212042316)

[3.2 SBOM Elements 11](#_Toc212042317)

[**3.2.1 Verification and reference material** 13](#_Toc212042318)

[**3.2.2 Rationale** 13](#_Toc212042319)

[3.3 File Format 13](#_Toc212042320)

[**3.3.1 Verification and reference material** 13](#_Toc212042321)

[**3.3.2 Rationale** 13](#_Toc212042322)

[3.4 Timing of SBOM Document delivery 14](#_Toc212042323)

[**3.4.1 Verification and reference material** 14](#_Toc212042324)

[**3.4.2 Rationale** 14](#_Toc212042325)

[3.5 Method of SBOM Document delivery 14](#_Toc212042326)

[**3.5.1 Verification and reference material** 14](#_Toc212042327)

[**3.5.2 Rationale** 14](#_Toc212042328)

[3.6 SBOM Document Scope 14](#_Toc212042329)

[**3.6.1 Verification and reference material** 15](#_Toc212042330)

[**3.6.2 Rationale** 15](#_Toc212042331)

[3.7 SBOM Document Verification 15](#_Toc212042332)

[**3.7.1 Verification and reference material** 15](#_Toc212042333)

[**3.7.2 Rationale** 15](#_Toc212042334)

[3.8 SBOM Document Confidentiality 15](#_Toc212042335)

[**3.8.1 Verification and reference material** 15](#_Toc212042336)

[**3.8.2 Rationale** 16](#_Toc212042337)

[4. Conformant notice 16](#_Toc212042338)

[5. SBOM Quality Assessment and Improvement Measures 16](#_Toc212042339)

[5.1 Ensuring Accurate and Consistent “Value” Information 16](#_Toc212042340)

[**5.1.1 Issue Overview** 16](#_Toc212042341)

[**5.1.2 Detailed Description** 16](#_Toc212042342)

[**5.1.3 Improvement Measures** 17](#_Toc212042343)

[**5.1.4 Risks and Considerations** 17](#_Toc212042344)

[5.2 Standardization and Normalization of Component Granularity 17](#_Toc212042345)

[**5.2.1 Issue Overview** 17](#_Toc212042346)

[**5.2.2 Detailed Description** 18](#_Toc212042347)

[**5.2.3 Improvement Measures** 18](#_Toc212042348)

[**5.2.4 Risks and Considerations** 18](#_Toc212042349)

[5.3 Complementing Source Code Information and Enhancing Transparency 19](#_Toc212042350)

[**5.3.1 Issue Overview** 19](#_Toc212042351)

[**5.3.2 Detailed Description** 19](#_Toc212042352)

[**5.3.3 Improvement Measures** 19](#_Toc212042353)

[**5.3.4 Risks and Considerations** 19](#_Toc212042354)

[5.4 Inconsistent Component Identification leading to Inadequate Vulnerability Handling 20](#_Toc212042355)

[**5.4.1 Issue Overview** 20](#_Toc212042356)

[**5.4.2 Detailed Description** 20](#_Toc212042357)

[**5.4.3 Improvement Measures** 21](#_Toc212042358)

[**5.4.4 Risks and Considerations** 21](#_Toc212042359)

[5.5 Enhanced Information Integration and Collaboration Between Upstream and Downstream 22](#_Toc212042360)

[**5.5.1 Issue Overview** 22](#_Toc212042361)

[**5.5.2 Detailed Description** 22](#_Toc212042362)

[**5.5.3 Improvement Measures** 22](#_Toc212042363)

[**5.5.4 Risks and Considerations** 22](#_Toc212042364)

[5.6 Establishing a Tamper Detection and Change Management System 22](#_Toc212042365)

[**5.6.1 Issue Overview** 22](#_Toc212042366)

[**5.6.2 Detailed Description** 22](#_Toc212042367)

[**5.6.3 Improvement Measures** 23](#_Toc212042368)

[**5.6.4 Risks and Considerations** 23](#_Toc212042369)

[5.7 Clarifying the Scope of Descriptions and Defining Accountability 23](#_Toc212042370)

[**5.7.1 Issue Overview** 23](#_Toc212042371)

[**5.7.2 Detailed Description** 24](#_Toc212042372)

[**5.7.3 Improvement Measures** 24](#_Toc212042373)

[**5.7.4 Risks and Considerations** 25](#_Toc212042374)

[5.8 Unified Expression of Inter-Component Relationships 25](#_Toc212042375)

[**5.8.1 Issue Overview** 25](#_Toc212042376)

[**5.8.2 Detailed Description** 25](#_Toc212042377)

[**5.8.3 Improvement Measures** 25](#_Toc212042378)

[**5.8.4 Risks and Considerations** 25](#_Toc212042379)

[5.9 Capturing Dependencies and Issues in Package Retrieval Methods 25](#_Toc212042380)

[**5.9.1 Issue Overview** 25](#_Toc212042381)

[**5.9.2 Detailed Description** 25](#_Toc212042382)

[**5.9.3 Improvement Measures** 25](#_Toc212042383)

[**5.9.4 Risks and Considerations** 25](#_Toc212042384)

[5.10 Insufficient Interoperability and Flexibility Among Tools 26](#_Toc212042385)

[**5.10.1 Issue Overview** 26](#_Toc212042386)

[**5.10.2 Detailed Description** 26](#_Toc212042387)

[**5.10.3 Improvement Measures** 26](#_Toc212042388)

[**5.10.4 Risks and Considerations** 26](#_Toc212042389)

[6. Compliance Elements for SPDX and CycloneDX in Relation to Legislation and Guidelines 27](#_Toc212042390)

[6.1 NTIA - The Minimum Elements For a Software Bill of Materials (SBOM) 27](#_Toc212042391)

[**6.1.1 Required Elements** 27](#_Toc212042392)

[**6.1.2 References** 28](#_Toc212042393)

[6.2 CISA - 2025 Minimum Elements for a Software Bill of Materials (SBOM) 29](#_Toc212042394)

[**6.2.1 Required Elements** 29](#_Toc212042395)

[**6.2.2 References** 29](#_Toc212042396)

[6.3 BSI TR-03183 Cyber Resilience Requirements for  Manufacturers and Products Part 2: Software Bill of Materials (SBOM) Version 2.1.0 29](#_Toc212042397)

[**6.3.1 Required Elements** 29](#_Toc212042398)

[**6.3.2 References** 30](#_Toc212042399)

[6.4 OpenChain Telco SBOM Guide Version 1.1 30](#_Toc212042400)

[**6.4.1 Required Elements** 30](#_Toc212042401)

[**6.4.2 References** 31](#_Toc212042402)

[Appendices 32](#_Toc212042403)

[Appendix.1 SBOM Samples 32](#_Toc212042404)

[Appendix.2 Authors 33](#_Toc212042405)

[Appendix.3 Version History 33](#_Toc212042406)

This page intentionally left blank

## Introduction

xxx

This guide is licensed under Creative Commons Attribution License 4.0 (CC-BY-4.0).

This page intentionally left blank

# **0. Preface**

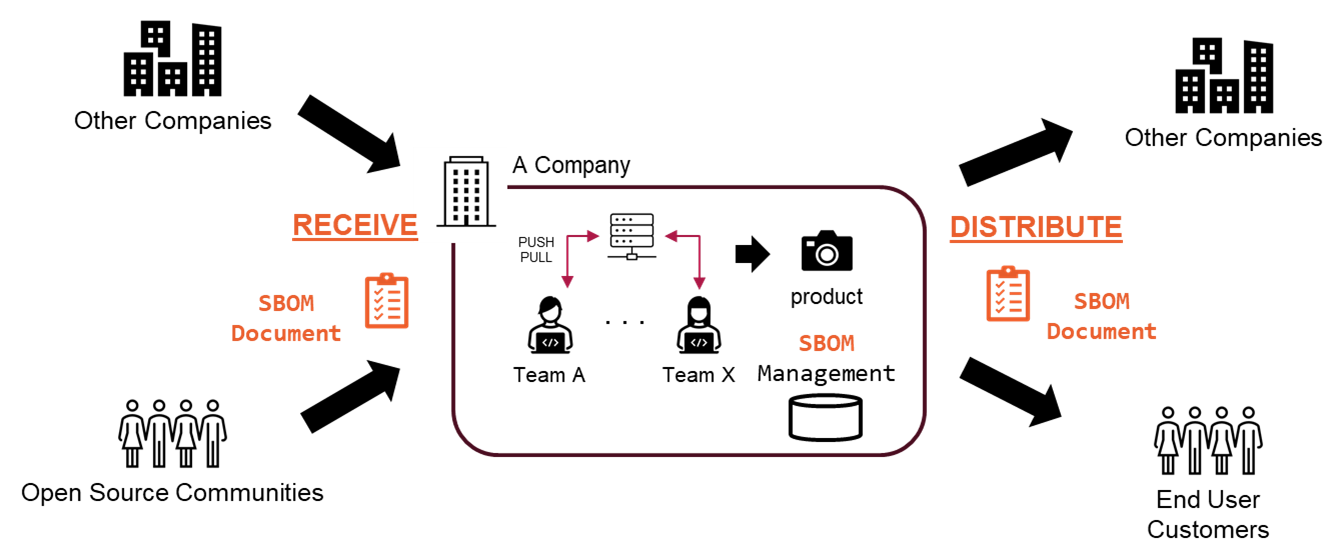
This guide, “OpenChain SBOM Document Quality Guide”, extends the “[OpenChain Telco SBOM Guide](https://github.com/OpenChain-Project/Telco-WG/)” by establishing a clear framework for document quality – centered on security assurance and license compliance – and providing actionable requirements to achieve it.

Key considerations and differences when adapting the Telco SBOM Guide to develop this guide:

* **Compatibility**: This guide is compatible with the Telco SBOM Guide. An SBOM Document conforming to the Telco SBOM Guide is also compliant with this guide.
* **Applicability**: While the Telco SBOM Guide is intended for cross-industry use, its wording has been refined to reflect applicability across various sectors.
* **Format Independence**: This guide is written to be independent of any specific SBOM Data format.
* **Quality Definition**: A new chapter discusses what constitutes a high-quality SBOM Document, explains its important, and describes how such documents can be effectively utilized.
* **Best practices**: Guidance addressing various challenges in creating and managing SBOM Documents have been incorporated.
* **Practical Examples**: As part of these best practices, practical SBOM Document samples are provided in JSON format along with their corresponding schema.

# **1. Scope and SBOM Document Quality**

While the term **"SBOM"** generally refers to the information that constitutes a software's composition, this guide specifically focuses on the quality of the “**SBOM Document”**. In this guide, **”SBOM Document”** is a structured artifact – typically formatted in JSON and based on specifications such as SPDX or CycloneDX – that is exchanged between software distributors and recipients.



This guide, “OpenChain SBOM Document Quality Guide”, establishes a clear framework for document quality – centered on security assurance and license compliance – and providing actionable requirements to achieve it.

Specifically, documents are evaluated based on following two essential aspects:

* Adequacy of Security Assurance  
  Assesses whether sufficient baseline information is provided to support an investigation that validates the software's security posture, even if, at the time of delivery, the document does not comprehensively cover all risks, vulnerabilities, or mitigation strategies.
* Effectiveness of License Compliance  
  Assesses whether the necessary licensing details and usage terms for each software component are properly captured to ensure compliance with relevant laws and regulations.

By adhering to this guide, stakeholders can ensure that the SBOM Documents exchanged within the software supply chain consistently meet high-quality standards.

# **2. Terms and Definitions**

Ongoing discussion at the following URL:

<https://docs.google.com/spreadsheets/d/1yQ3E-moYXpSA-7ov4SndXJW6Lx82wkz44GGU4jfESWE/edit?gid=0#gid=0>

<https://github.com/OpenChain-Project/OpenChain-JWG/discussions/415>

# **3. Guidelines to Enhance SBOM Document Quality**

Chapter 3 defines the criteria by which the OpenChain SBOM Document Quality Guide recognizes an SBOM Document as high-quality.

## **3.1. Data Format**

An OpenChain SBOM Document Quality Guide compatible document SHALL use a format generally recognized as the industry standard for SBOMs - such as SPDX® or CycloneDX.

### **3.1.1 Verification and reference material**

SPDX®: <https://spdx.dev/use/specifications/> , <https://tools.spdx.org/app/validate/>

CycloneDX: <https://cyclonedx.org/specification/overview/>

### **3.1.2 Rationale**

To ensure simplified handling and streamlining of tooling and competences in the software supply chain, both for suppliers and consumers of software, OpenChain SBOM Document Quality Guide Compatible documents shall employ a format generally recognized as the standard specification for SBOMs as standardized. By harmonizing the use of this standard SBOM Data Format in an organization's external interfaces, the reduction in complexity leads to fewer errors and, as a result, improved the document quality.

## **3.2 SBOM Elements**

An SBOM Document SHALL include all the SBOM Elements defined in this chapter, which the OpenChain SBOM Document Quality Guide considers indicative of high quality.

SBOM Document Information

* Data Format Version
* License for the SBOM Document
* Unique ID for the SBOM Document  
  Include a global unique identifier in the SBOM Document to represent the exchanged  SBOM Document. The specification for this unique value varies depending on the Data Format;   
  For example, CycloneDX has 'serialNumber', while in SPDX either the 'SPDXID' must be global unique or a combination of 'DocumentNamespace' and 'SPDXID' is used to ensure uniqueness.
* Creation information for the SBOM Document  
  It is essential to include the details of who created the SBOM Document and when, as this information is necessary for contacting the responsible entities in case of issues.
  + Author of SBOM Document  
    If permitted by the respective Data Format specifications, the creator’s information should ideally contain a valid email address or a URL for a contact form to ensure they are reachable.
  + Creation Date and Time of SBOM Document
  + SBOM Types  
    How and when each element value is recorded in an SBOM Document can be critical for effective configuration and vulnerability management. Therefore, including the [SBOM Types defined by CISA](https://www.cisa.gov/resources-tools/resources/types-software-bill-materials-sbom) in the document can enhance its overall quality. However, since there is no standardized element for SBOM Types, it is essential to harmonize their usage across the supply chain.

Package information

* Information that can uniquely identify a software package  
  For accurate license compliance and vulnerability assessment, it is essential to correctly identify each software package. The document should include the following elements to ensure proper identification:
  + Package Name
  + Package Version
  + Details regarding the package identifiers, such as the [SoftWare Hash IDentifiers (SWHID)](https://www.swhid.org/), the [package URL (PURL)](https://github.com/package-url/purl-spec), [CPE](https://nvd.nist.gov/products/cpe) or the URL of the package distribution site, indicating where the corresponding software package can be obtained.
* Package Supplier information  
  This element should contain contact details for the supplier of the package. It is desirable to include a valid email address or a URL to a contact form, ensuring that inquiries about the package can be addressed promptly.
* Component Hash value  
  Each software component listed in the SBOM Document should include at least one corresponding hash value from the distributed component. This is necessary to verify that the particular component is indeed present in the distribution. The hash algorithm must be chosen from those specified in the respective Data Format and used to calculate the hash of the corresponding source code or binary component.
* Proprietary Software indicator  
  For proprietary software, elements like Package Name and Package Version may be custom-defined. However, the SBOM Document should clearly indicate the software is proprietary - for example, using package comments or a purl - and should include Package Supplier information to facilitate inquiries.

License information

* License declared in the Package  
  The SBOM Document should include the license information declared by the distributed package. The declared license SHALL utilize standardized identifiers, for example, [the SPDX License Identifier](https://spdx.org/licenses/), to ensure clarity and consistency.
* License concluded by the Package Distributor  
  The SBOM Document should include the license determined by the package distributor. For example, when a package is dual-licensed, the relationship with other packages may dictate which license applies. In addition, considering both the declared license and the overall software structure, the distributor must conclude the  license for distribution. It is recommended that the distributor include such  Element in the SBOM Document. If the appropriate license cannot be concluded - and if allowed by the applicable specifications - the element may either be omitted or marked as a ‘known unknown’, using a placeholder such as ‘NOASSERTION’.

Relationship information

* As specified by NTIA SBOM Minimum Elements, the document must include at least the “DESCRIBES” and “CONTAINS” relationship types.

Mandatory information required by Specifications and Guidelines

* Other mandatory Elements as specified in each Data Format  
  Include all Elements that are mandatory for each Data Format. If the appropriate value for any element is unknown, fill it in with a placeholder such as  'NOASSERTION', indicating a 'known unknown’.  
  Since each Entity typically processes the SBOM Document using different tools, including these required Elements enhances interoperability and ultimately improves the quality of SBOM Documents across the entire supply chain.  
  Refer to [Appendix.X](https://docs.google.com/spreadsheets/d/1SuGv1L3H_-Iq6dmH7DnjDgAa90LCRnoHB3DTfuWh0Jg/edit?gid=1936044844#gid=1936044844) for more details.

* Other mandatory Elements as specified by the applicable specifications and guidelines that each Entity conforms  
  Include all Elements that are mandatory for each specification and guideline to which each Entity conforms. If the appropriate value for any element is unknown, fill it in with a placeholder such as  'NOASSERTION', indicating a 'known unknown'.  
  When a unified specification across the entire supply chain is not mandated, it is important to clearly indicate which specification the SBOM Document conforms to. This improves the document’s quality by explicitly defining the compliance requirements.

### **3.2.1 Verification and reference material**

The Minimum Elements For a Software Bill of Materials(SBOM): <https://www.ntia.gov/report/2021/minimum-elements-software-bill-materials-sbom>

SPDX®: <https://spdx.dev/use/specifications/>

CycloneDX: <https://cyclonedx.org/specification/overview/>

SWHID: <https://www.swhid.org/>

PURL: <https://github.com/package-url/purl-spec>

### **3.2.2 Rationale**

Clearly defining across the entire supply chain which elements, such as SBOM Document Creation information and Package information details, must be included and distributed helps prevent gaps in the distributed data. Furthermore, clarifying the expected values for each element reduces ambiguous language and inter-tool variations.

## **3.3 File Format**

An OpenChain SBOM Document Quality Guide Compatible document SHALL be in a machine-readable format, such as formats currently supported by SCA tools, including JSON-LD, JSON, XML, YAML, and the SPDX Tag:Value etc. In addition, to facilitate scenarios where a person may need to review the document (for example, for license verification),　the document SHOULD be provided in, or be easily convertible to, a human-readable format.

### **3.3.1 Verification and reference material**

XML 1.0: <https://www.w3.org/TR/xml/>JSON-LD 1.1: <https://www.w3.org/TR/json-ld11/>

ECMA-404: The JSON data interchange syntax: <https://ecma-international.org/publications-and-standards/standards/ecma-404/>

YAML™: <https://yaml.org/spec/>

SPDX®: 4.4 Standard data format requirements: <https://spdx.github.io/spdx-spec/v2.3/conformance/#44-standard-data-format-requirements>

For other formats, please refer to the respective specifications.

### **3.3.2 Rationale**

As noted in NTIA SBOM Minimum Elements, when SBOM Documents are exchanged across the supply chain, they must be provided in a machine-readable standard format to ensure interoperability among the various tools used by different entities. At the same time, since SBOM Documents may also be reviewed manually (for example, for license verification), it is equally important that the document is available in, or can be easily converted to, a human-readable format.

## **3.4 Timing of SBOM Document delivery**

The SBOM Document SHALL be delivered no later than at the time of the delivery of the software (in either binary or source form).

### **3.4.1 Verification and reference material**

The Minimum Elements For a Software Bill of Materials(SBOM), “Distribution and Delivery”: <https://www.ntia.gov/report/2021/minimum-elements-software-bill-materials-sbom>

### **3.4.2 Rationale**

The SBOM Document may be provided before the software delivery if desired, but a corresponding SBOM Document shall be delivered no later than at the delivery of the software to ensure that the receiving entity can ingest the software and its SBOM.

## **3.5 Method of SBOM Document delivery**

The SBOM Document SHALL be attached to the package or made available via a networked resource. If provided over a network, it is desirable that recipients have access to it for a defined period (e.g. 12 months as specified by the [OpenChain recertification requirements](https://github.com/OpenChain-Project/Reference-Material/blob/master/OpenChain-Standards-Self-Certification/Checklist/ISO-IEC-5230/en/iso-5230-2020-Self-Certification-Checklist-1.2.md#section-6-adherence-to-the-specification-requirements)). In all cases, authorized users SHALL promptly access, copy, and locally store the document, and the delivery method SHOULD clearly link it to the specific software version (e.g. via a version-specific URL).

### **3.5.1 Verification and reference material**

The Minimum Elements For a Software Bill of Materials(SBOM), “Distribution and Delivery”: <https://www.ntia.gov/report/2021/minimum-elements-software-bill-materials-sbom>

### **3.5.2 Rationale**

SBOM Document delivery such as webhosting are less stable and access is not guaranteed over time; however “embedding” may not be technically feasible. Thus, in scenarios where it is not possible on technical grounds to include the SBOM Document in the software delivery, publishing the SBOM Document online is permitted.

## **3.6 SBOM Document Scope**

The SBOM SHALL contain all open source software that is delivered with the product including all of the transitive dependencies. The SBOM SHOULD contain all commercial components.  
If some components are not included, they SHALL be reported as “Known Unknowns”.  
Refer to the “[5.7 Clarifying the Scope of Descriptions and Defining Accountability](https://docs.google.com/document/d/1iuXX8j10N70dfce1-CZFWhW6S2jEqc--flcCgXMMdjg/edit?tab=t.0#heading=h.8tf25jdzscvd)” for more details.

### **3.6.1 Verification and reference material**

The Minimum Elements For a Software Bill of Materials(SBOM), “Known Unknowns”: <https://www.ntia.gov/report/2021/minimum-elements-software-bill-materials-sbom>

### **3.6.2 Rationale**

It might not be possible, advisable or feasible to have the commercial component information in the SBOM Document. However, the SBOM Document should convey as complete information as possible.

## **3.7 SBOM Document Verification**

A digital signature of the SBOM Document SHOULD be provided in order to guarantee its integrity.

### **3.7.1 Verification and reference material**

Sigstore: <https://www.sigstore.dev/>

### **3.7.2 Rationale**

While the need to attach a digital signature to the SBOM Document for ensuring its integrity is currently under discussion, few practical implementations exist. Consequently, this topic  will be revisited in the future, taking into account the associated costs and practical considerations.

## **3.8 SBOM Document Confidentiality**

Contents of the SBOM Document may be subject to confidentiality agreements. Appropriate access controls SHOULD be implemented as necessary. However, confidentiality agreements and access controls SHALL NOT prevent a recipient from redistributing the SBOM Document along with the applicable software components.

### **3.8.1 Verification and reference material**

The Minimum Elements For a Software Bill of Materials(SBOM), “Access Control”: <https://www.ntia.gov/report/2021/minimum-elements-software-bill-materials-sbom>

### **3.8.2 Rationale**

Information related to proprietary software may be subject to confidentiality agreements and thus cannot always be public. In such cases, it is acceptable either to restrict access solely to authorized entities within the supply chain or to substitute the information with "Known Unknowns."

However, when these components are distributed through complex supply chains, confidentiality agreements and access controls SHALL NOT restrict SBOM Document redistribution, thereby preventing the loss of critical information during the distribution.

# **4. Conformant notice**

You MAY use the following statement to indicate that the software components you distribute are accompanied by SBOM Document conforming to the OpenChain SBOM Document Quality Guide:

**“This software is provided with SBOM Document defined as high quality by the OpenChain SBOM Quality Guide, which is available at** [**https://github.com/OpenChain-Project/XXX**](https://github.com/OpenChain-Project/XXX)”

# **5. SBOM Quality Assessment and Improvement Measures**

This section highlights the challenges often encountered when generating and managing SBOM Document, and introduces best practices for addressing them. These measures are designed to enhance the accuracy, consistency, and transparency of SBOM Document, as well as to improve the overall processes involved in handling them.

## **5.1 Ensuring Accurate and Consistent “Value” Information**

### **5.1.1 Issue Overview**

Challenges exist in the inconsistent representation of information such as package names, versions, and supplier names across different companies and tools. Without unified standards, automatic analysis of SBOM Document or vulnerability matching becomes challenging, leading to inaccuracies.

### **5.1.2 Detailed Description**

In many SBOM guides and standards, the element names defining what should be included are explicitly specified along with the corresponding value ranges and representation methods. However, since a precise format for these values is often not defined, issues can arise during practical implementation.

For example, the two SBOMs below use different package names - one as 'hello' and the other as 'hello 0.0.1'. This discrepancy is due to differences in the output formats of the tools used.

* [example7-bin.spdx.json](https://github.com/spdx/spdx-examples/blob/master/software/example7/spdx2.2/example7-bin.spdx.json#L44)"name": "hello",
* [hello-dist.spdx.json](https://github.com/spdx/spdx-examples/blob/master/software/example12/spdx2.2/hello-dist.spdx.json#L56)"name": "hello 0.0.1",

Although verifying that these represent the same package is possible – for example, by comparing their PURLs – the fact that different tools may output different values for the same element name frequently leads to confusion and poses challenges to the smooth operation of SBOM management.

### **5.1.3 Improvement Measures**

* The SBOM Document shall include the items documented in "3.2 SBOM Elements - Package information, Information that can uniquely identify a software package”, ensuring their consistent use throughout the entire supply chain.
* Additionally, since the identifiers used to uniquely identify a package may differ depending on the entity distributing the SBOM Document, it is necessary to also provide information on which combination of elements and corresponding values should be used to accurately identify the software package.

### **5.1.4 Risks and Considerations**

* If the documentation rules are inadequately defined or overly complex, there is a risk of misclassification due to incomplete handling of exceptional cases or unique notations.
* If the rules deviate from actual operational practices, there is a risk of misclassification resulting from either insufficient or excessively stringent checks.
* It should be noted that complete automation of tools and processes is challenging; therefore, final checks and exception handling will require manual review.

## **5.2 Standardization and Normalization of Component Granularity**

### **5.2.1 Issue Overview**

Exchanged SBOM Documents may define components exclusively at the file level, exclusively at the package level, or include a mix of both. This variability in granularity can lead to challenges when describing dependencies consistently across the supply chain.

### **5.2.2 Detailed Description**

This challenge is illustrated using a simplified scenario involving three parties: **Vendor**, **Maker**, and **User**.

The assumptions are as follows:

* Vendor provides ‘application A‘ binary along with SBOM Document to Maker
* Maker supplies ‘product X‘ (which includes application A and other OSS) along with SBOM Document to User
* ‘application A‘ depends on the OSS used in product X.

Under these conditions, the following table summarizes the possible combinations of component granularity in the SBOM Document.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | SBOM Document for ‘product X’ | SBOM Document for ‘product X’ |
|  | component granularity | file-level | package-level |
| SBOM Document for ‘application A’ | file-level | All dependency shall be recorded and provided at the file-level. | **Maker** need to convert file-level information to package-level for application A. |
| SBOM Document for ‘application A | package-level | **Maker** need to decompose package-level information and convert them to  file-level. | All dependency shall be recorded and provided at the package-level. |

Differences in component granularity within SBOM Document require a full review to determine its granularity level, leading to an increased workload for the integrating entity.

### **5.2.3 Improvement Measures**

* Ensure that component granularity is clearly indicated throughout the supply chain to enable appropriate handling based on that granularity.
* Define explicit granularity elements in future versions of the Data format specifications to clarify whether the information is at the file or package level.

### **5.2.4 Risks and Considerations**

* There is a potential for issues when linking with vulnerability information. For example, if component granularity is mixed, it may not be possible to automatically identify the corresponding vulnerability information, leading to concerns over increased manual effort and time.

## **5.3 Complementing Source Code Information and Enhancing Transparency**

### **5.3.1 Issue Overview**

When components are exchanged in binary form, source code information may be needed to verify licensing and check for vulnerabilities. Providing source code details in the SBOM Document can enhance supply chain transparency.

### **5.3.2 Detailed Description**

Adhering to "3.2 SBOM Elements – Package information, Information that can uniquely identify a software package" can sometimes allow you to locate source code in repositories like GitHub. However, when creating binary components, patches are often applied for customization or to fix vulnerabilities and bugs. In such cases, it becomes challenging to pinpoint the exact source code used. Furthermore, even for the same component and version, differences in SBOM types can lead to discrepancies between the source code included in the delivered binary component and the source code provided. Therefore, it is necessary to retain the exact source code information used during the build that is included in the binary component.

### **5.3.3 Improvement Measures**

To improve supply chain transparency, the SBOM Document author can attach source code information for binary components as follows:

* Add the source code information for a binary component as a separate source code component in the SBOM Document and manage its details. For example, SPDX allows linking binary components and source code components using relationships such as GENERATED\_FROM, DESCENDANT\_OF, or CONTAINS (see also Section 5.8).
* CycloneDX provides a “pedigree” attribute for holding source code details, allowing the SBOM Document to include direct source code and patch information for the binary component.

The source code information should include, when available:

* The source URL (this may be the URL of a tar.xz file or a version control system URL).
* A hash value of the source code (for instance, the SHA-512 hash for a tar.xz file or a commit hash for a version control system; SWHID can also be used).
* If modifications were made during the build process, corresponding patch information should be included alongside the source code details.

### **5.3.4 Risks and Considerations**

Verifying the validity of the source code information attached by the SBOM Document provider to binary components is challenging for SBOM Document recipients.

One potential method of verification is for the recipient to rebuild the binary and confirm that it matches. However, to do so, the SBOM Document provider must supply additional build environment information and ensure that the build is reproducible.

## **5.4 Inconsistent Component Identification leading to Inadequate Vulnerability Handling**

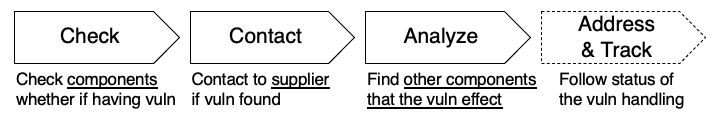
### **5.4.1 Issue Overview**

SBOM is expected to enhance vulnerability handling by providing:

* Component identification information that can be cross-referenced with vulnerability databases to determine if a vulnerability exists
* Supplier information for reporting to remediate found vulnerabilities
* Information of inter-component dependencies to assess how a vulnerability in one component might affect others

However, these elements are not always captured accurately or comprehensively, leading to:

* Inconsistent naming and identifier usage that hampers effective correlation with vulnerability data
* Inadequate or ambiguous supplier information, reducing the ability to follow up on vulnerability reports
* Incomplete description of dependency relationships, risking oversight of cross-component impacts



In addition, detailed information and status of vulnerabilities, if available, can be also described in formats such as VEX (Vulnerability Exploitability eXchange) and associated with SBOM to be used for their remediation and tracking of changes.

### **5.4.2 Detailed Description**

1. When relying on component names and version numbers for identification, inconsistencies such as label variations (as highlighted in Section 5.1) can occur, leading to missed detection of vulnerabilities. Even the inclusion of unique identifiers like PURLs or CPE names does not guarantee a match in vulnerability databases, since those identifiers may not be present there.
2. SBOM Document generated by automated tools often fall short in providing thorough supplier information. The supplier fields may be omitted or described in a way that does not clearly identify the organization or individual responsible – for instance, sometimes only a name is provided without accompanying contact information such as an email address. This lack of precise supplier data complicates vulnerability reporting and follow-up actions.
3. The completeness of inter-component dependency information depends heavily on the tools, settings, and the technologies (programming languages or development environments) used. This can result in an incomplete mapping of dependency relationships, which in turn may lead to an underestimation of how vulnerabilities in one component can impact others within the software product.

This detailed breakdown underscores that despite SBOM Document’s potential in improving vulnerability handling, issues with inconsistent component identification and incomplete data often limit its effectiveness.

### **5.4.3 Improvement Measures**

1. Refer to Section 5.1 and use the naming conventions employed by vulnerability databases such as Open Source Vulnerabilities (OSV) and National Vulnerability Database (NVD) to record the component name, and include the corresponding unique software identifiers (e.g., a PURL for OSV or a CPE name for NVD) within the component information.
2. Ensure that the supplier information identifies a real, publicly recognized entity or individual by verifying its existence and providing clear contact details such as the name, a valid email address, or a URL.
3. Utilize tools that can extract inter-component dependency information from package managers to generate a comprehensive SBOM Document, and when possible, use build tools capable of producing a Build SBOM according to CISA’s SBOM Type Classification.

### **5.4.4 Risks and Considerations**

1. Although unique identifiers such as PURLs or CPE names can assist with component identification, their support is limited because only some databases like OSV support PURLs while others like NVD may not, and different databases might list the same component using slightly varying CPE names. Moreover, dynamically loaded components at runtime may not be captured in the SBOM, which further compromises the accuracy of vulnerability correlation.
2. Since a component's supplier may change over time due to mergers, business closures, or shifts in support responsibilities, it is essential to verify and update supplier information on a regular basis.
3. Automatically extracted dependency information, especially when derived from static analysis or less comprehensive techniques rather than robust build tools, may not accurately capture the actual runtime inter-component relationships, particularly in cases where components are dynamically loaded, leading to gaps in the overall depiction of dependencies within the SBOM Document.

## **5.5 Enhanced Information Integration and Collaboration Between Upstream and Downstream**

### **5.5.1 Issue Overview**

Discrepancies between SBOMs provided by upstream vendors and those generated internally by integrators can lead to misalignment and integration challenges.

### **5.5.2 Detailed Description**

### **5.5.3 Improvement Measures**

### **5.5.4 Risks and Considerations**

## **5.6 Establishing a Tamper Detection and Change Management System**

### **5.6.1 Issue Overview**

In SBOM operations, the SBOM Document may be updated on a different schedule than the  provided software itself, and its content may contain errors. However, there is currently no sufficient mechanism to detect modifications to the SBOM Document, making it difficult to guarantee consistency with the software.

### **5.6.2 Detailed Description**

* Manual modifications leading to data inconsistencies  
  When the SBOM Document is later edited or supplemented by hand, the resulting document may no longer match the original software configuration, risking unintended changes such as the addition of unneeded component information.
* Lack of version control causing opaque modification history  
  Without a system to record changes, it becomes impossible to trace who modified the SBOM Document, when, why, or how, making it difficult to distinguish between intentional tampering and routine updates, and preventing rollback to previous, trusted versions.
* Inconsistent formats and missing information  
  When SBOM Documents are automatically generated by multiple tools, differences in format and content can result in the loss of critical details – such as dependency information – during integration or editing, potentially leading to incomplete security assessments.

### **5.6.3 Improvement Measures**

* Digital signatures  
  Immediately append a digital signature or hash value to the SBOM upon generation to enable later detection of any modifications.
* Change history management  
  Record and track all changes to the SBOM file using a version control system.
* Automated verification  
  Implement tools that automatically check the integrity of digital signatures or hash values.
* Automated monitoring  
  Establish a tamper-detection process that regularly verifies signatures and issues alerts when abnormalities are detected.
* Update and redistribution policies  
  Formalize procedures for regularly updating the SBOM, reapplying signatures, and automatically redistributing the revised document.
* Audit and review framework  
  Complement automated measures with periodic reviews, log audits, and manual final verifications.

### **5.6.4 Risks and Considerations**

* Key management risks  
  Be alert to the leakage or expiration of signing keys and certificates; implement proper key management and revocation procedures.
* Increased operational burden  
  Introducing new processes may necessitate changes to development environments and incur additional team training costs.
* Inter-tool compatibility issues  
  Differences in SBOM tool formats can compromise the consistency of hash values and signatures.
* Minimizing manual intervention  
  Maximize automation to eliminate human errors, limiting manual verification to the minimum required.

## **5.7 Clarifying the Scope of Descriptions and Defining Accountability**

### **5.7.1 Issue Overview**

SBOM documents largely rely on the provider's discretion, resulting in inconsistent comprehensiveness and accuracy of the required information. Consequently, essential data may be missing across the software supply chain, increasing the risk of confusion in managing security and license compliance.

### **5.7.2 Detailed Description**

Various guidelines and specifications exist, yet the criteria for including dependent components in an SBOM and the level of detail required for each component remain unclear.

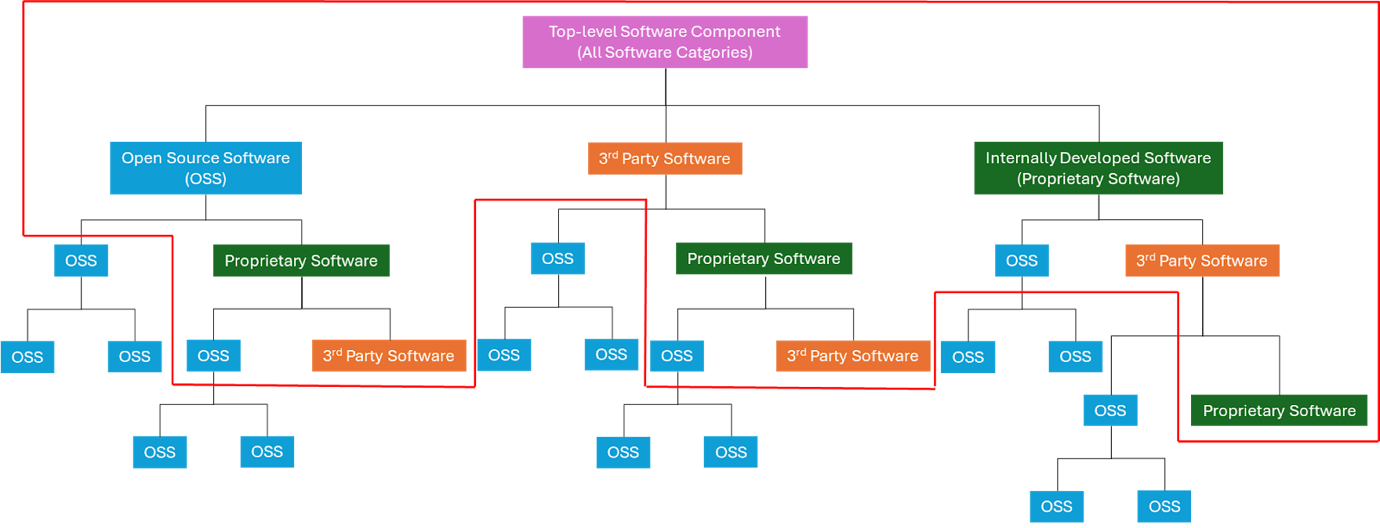
* Only a limited set of necessary details, such as the scope of software dependencies and version information, may be included.
* It is often unclear who is accountable for each component, as contact information and management responsibilities are not clearly defined. This lack of clarity hampers rapid response when issues arise.

Consequently, incomplete information within the SBOM can prevent a full understanding of vulnerabilities and license-related risks across the entire software supply chain, potentially impeding effective security measures.

### **5.7.3 Improvement Measures**

* Establish clear, standardized criteria for the items that need to be included in an SBOM across the entire software supply chain, with particular emphasis on clarifying the scope of dependencies.
* Clearly define accountability for each item by including specific responsible individuals and contact details within the document.

This approach will enable effective information sharing and rapid response when issues arise.



More practically, software distributors in principle have access only to the detailed information contained in the red-framed area in the figure above. Accordingly, they should assume at least a minimal accountability for each software component within this range.

However, for Open Source Software (OSS), obtaining sufficient upstream information may not always be possible. In such cases, any unavailable details should be explicitly marked as 'Known Unknown,' while incorporating as much of the available OSS information as possible into the SBOM Document.

### **5.7.4 Risks and Considerations**

* Delays in updating the SBOM Document in response to software version upgrades or changes in dependencies may result in decisions being made based on outdated information.
* A lack of uniform criteria across the supply chain can lead to inconsistencies among SBOM Documents, causing confusion.
* If the responsible parties or contact information are not clearly specified or are recorded incorrectly, the ability to respond swiftly during issues may be compromised, thereby increasing security risks.

## **5.8 Unified Expression of Inter-Component Relationships**

### **5.8.1 Issue Overview**

Inconsistent depiction of relationships between components (e.g., dependency, containment, derivation) may hinder accurate automated analysis and risk assessment.

### **5.8.2 Detailed Description**

### **5.8.3 Improvement Measures**

### **5.8.4 Risks and Considerations**

## **5.9 Capturing Dependencies and Issues in Package Retrieval Methods**

### **5.9.1 Issue Overview**

Disparities between dependencies extracted directly from source code versus those derived from package management systems can lead to redundant or erroneous information.

### **5.9.2 Detailed Description**

### **5.9.3 Improvement Measures**

### **5.9.4 Risks and Considerations**

## **5.10 Insufficient Interoperability and Flexibility Among Tools**

### **5.10.1 Issue Overview**

Different SBOM generation and analysis tools, such as ORT (OSS Review Toolkit), FOSSology, or ScanCode, may produce outputs with variations that complicate integration in a unified processing workflow.

### **5.10.2 Detailed Description**

Many software tools designed for generating and managing SBOM Document are typically developed with interoperability in mind. However, in practical settings, there have been few comprehensive evaluations of the compatibility across these various tools, rendering their true degree of interoperability uncertain.   
Moreover, even when compatibility exists, there is generally insufficient documentation on the integration process, which consequently forces users to resort to trial-and-error methods to achieve effective configurations.

### **5.10.3 Improvement Measures**

When employing a mixed environment of tools for handling SBOM Documents, one can expect an increase in manual adjustments such as value conversions and schema modifications within the SBOM Document. To alleviate these issues, it is effective to designate a primary format to be used at the operational level.

The evaluation criteria for selecting a suitable format include the following:

* Existing assets
* The extent of compatibility across a wide range of tools
* The proficiency of the personnel responsible

Additionally, it is important to note that SBOM Documents received from upstream sources should retain identifiers—for example, purl—and generator information for precise software identification. In the continuous integration environment, only syntactic validation (such as with JSON Schema) is intended to be automated for the time being. Procedures for format transitions should be standardized through configuration files and minimal sample setups, with exception handling fixed according to established procedures. This approach enables gradual expansion while minimizing integration burdens and errors. Given that specifications and version differences are subject to change, it is essential to verify the applicable details against primary sources at the time of implementation.

### **5.10.4 Risks and Considerations**

# **6. Compliance Elements for SPDX and CycloneDX in Relation to Legislation and Guidelines**

This chapter provides a comprehensive analysis of the compliance elements associated with various guidelines within the frameworks of SPDX and CycloneDX. It is organized into sections that detail each guideline's requirements, categorizing them as mandatory or recommended. For each guideline, the chapter outlines the key elements, including their names, associated data fields, and supplementary remarks. Emphasis is placed on establishing a clear mapping between the guideline elements and the corresponding data fields defined in SPDX and CycloneDX. This structured approach facilitates cross-referencing and ensures consistency in documenting compliance requirements across multiple frameworks.

## **6.1 NTIA - The Minimum Elements For a Software Bill of Materials (SBOM)**

<https://www.ntia.gov/report/2021/minimum-elements-software-bill-materials-sbom>

### **6.1.1 Required Elements**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement Level | Element Name | SPDX v2.2+ | SPDX v3.0 | CycloneDX v1.6 |
| SHALL | Supplier Name | (7.5) Package Supplier | Core/Classes/Artifact.suppliedBy | bom.metadata.supplier,  bom.components[].supplier |
| SHALL | Component Name | (7.1) Package Name | Software/Classes/Package.name inherited from Core/Classes/Element.name | bom.components[].name |
| SHALL | Version of the Component | (7.3) Package Version | Software/Classes/Package.packageVersion | bom.components[].version |
| SHALL | Other Unique Identifiers | (7.2) Package SPDX Identifier  (6.5) SPDX Document Namespace  — (7.21) External Reference field | Software/Classes/SoftwareArtifact.contentIdentifier for SPDX Software Artifacts or Software/Classes/Package.packageUrl if the packageUrl is considered to be unique,  or Core/Classes/Element.externalIdentifier for resources outside the scope of SPDX-3.0 content | bom.components[].cpe,purl,swid |
| SHALL | Dependency Relationship | (11.1) Relationship: CONTAINS, DESCRIBES  The document must DESCRIBES at least one package. | Core/Classes/Relationship | bom.dependencies[] |
| SHOULD | Author of SBOM Data | (6.8) Creator | Core/Classes/CreationInfo.createdBy | bom.metadata.author |
| SHOULD | Timestamp | (6.9) Created | Core/Classes/CreationInfo.created | bom.metadata.timestamp |
| SHOULD | Hash of the Component | (7.10) Package Checksum | Core/Classes/Element.verifiedUsing | bom.components[].hashes[] |
| SHOULD | Lifecycle Phase | N/A —  (6.10) Creator comment | Software/Sbom/sbomType | bom.metadata.lifecycles[] |
| SHOULD | Other Component Relationships | (11.1) Relationship: GENERATES, ANCESTOR\_OF, VARIANT\_OF etc. | Core/Classes/Relationship | bom.components[].pedigree |
| SHOULD | License Information | (7.13) Concluded License  (7.15) Declared License | Core/Classes/Relationship  RelationshipType::hasConcludedLicense  RelationshipType::hasDeclaredLicense | bom.components[].licenses[] |

### **6.1.2 References**

SPDX v2.2+:

<https://spdx.github.io/spdx-spec/v2.3/how-to-use/#k2-satisfying-ntia-minimum-elements-for-an-sbom-using-spdx>

SPDX v3.0:

<https://github.com/spdx/using/blob/main/docs/comply-with-norms.md#satisfying-ntia-minimum-elements-for-an-sbom-using-spdx--us-executive-order-14028>

CycloneDX v1.6:

<https://cyclonedx.org/guides/OWASP_CycloneDX-Authoritative-Guide-to-SBOM-en.pdf>

## **6.2 CISA - 2025 Minimum Elements for a Software Bill of Materials (SBOM)**

<https://www.cisa.gov/resources-tools/resources/2025-minimum-elements-software-bill-materials-sbom>

### **6.2.1 Required Elements**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement Level | Element Name | SPDX v2.2+ | SPDX v3.0 | CycloneDX v1.6 |
| SHALL |  |  |  |  |
| SHALL |  |  |  |  |
| SHALL |  |  |  |  |
| SHALL |  |  |  |  |
| SHALL |  |  |  |  |
| SHOULD |  |  |  |  |
| SHOULD |  |  |  |  |
| SHOULD |  |  |  |  |
| SHOULD |  |  |  |  |
| SHOULD |  |  |  |  |
| SHOULD |  |  |  |  |

### **6.2.2 References**

## **6.3 BSI TR-03183 Cyber Resilience Requirements for  Manufacturers and Products Part 2: Software Bill of Materials (SBOM) Version 2.1.0**

<https://www.bsi.bund.de/EN/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/Technische-Richtlinien/TR-nach-Thema-sortiert/tr03183/TR-03183_node.html>

### **6.3.1 Required Elements**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement Level | Element Name | SPDX v2.2+ | SPDX v3.0 | CycloneDX v1.6 |
| SHALL |  |  |  |  |
| SHALL |  |  |  |  |
| SHALL |  |  |  |  |
| SHALL |  |  |  |  |
| SHALL |  |  |  |  |
| SHOULD |  |  |  |  |
| SHOULD |  |  |  |  |
| SHOULD |  |  |  |  |
| SHOULD |  |  |  |  |
| SHOULD |  |  |  |  |
| SHOULD |  |  |  |  |

### **6.3.2 References**

## **6.4 OpenChain Telco SBOM Guide Version 1.1**

<https://github.com/OpenChain-Project/Telco-WG/blob/main/OpenChain-Telco-SBOM-Guide_EN.md>

### **6.4.1 Required Elements**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement Level | Element Name | SPDX v2.2+ | SPDX v3.0 | CycloneDX v1.6 |
|  | Document Creation Information |  |  |  |
| SHALL | SPDXVersion: mandatory in SPDX | (6.1) SPDX Version |  |  |
| SHALL | DataLicense: mandatory in SPDX | (6.2) Data License |  |  |
| SHALL | SPDXID: mandatory in SPDX | (6.3) SPDX Identifier |  |  |
| SHALL | DocumentName: mandatory in SPDX | (6.4) Document Name |  |  |
| SHALL | DocumentNamespace: mandatory in SPDX | (6.5) SPDX Document Namespace |  |  |
| SHALL | Creator: mandatory in SPDX | (6.8) Creator |  |  |
| SHALL | Created: mandatory in SPDX | (6.9) Created |  |  |
| SHALL | CreatorComment: to be able to put “SBOM Build information” | (6.10) Creator Comment |  |  |
|  | Package Information |  |  |  |
| SHALL | PackageName: mandatory in SPDX | (7.1) Package Name |  |  |
| SHALL | SPDXID: mandatory in SPDX | (7.2) Package SPDX Identifier |  |  |
| SHALL | PackageVersion: needed by “NTIA SBOM Minimum elements” | (7.3) Package Version |  |  |
| SHALL | PackageSupplier: needed by “NTIA SBOM Minimum elements” | (7.5) Package Supplier |  |  |
| SHALL | PackageDownloadLocation: mandatory in SPDX | (7.7) Package Download Location |  |  |
| SHALL | PackageLicenseConcluded: mandatory in SPDX 2.2 | (7.13) Concluded License |  |  |
| SHALL | PackageLicenseDeclared: mandatory in SPDX 2.2 | (7.15) Declared License |  |  |
| SHALL | PackageCopyrightText: mandatory in SPDX 2.2 | (7.17) Copyright Text |  |  |
| SHALL | Relationship: at least DESCRIBES and CONTAINS, needed by “NTIA SBOM Minimum elements” | (11.1) Relationship: CONTAINS, DESCRIBES |  |  |
| SHOULD | One of the two attributes **PackageChecksum** or PackageVerificationCode is RECOMMENDED: recommended by “NTIA SBOM Minimum elements” | (7.10) Package Checksum |  |  |
| SHOULD | One of the two attributes PackageChecksum or **PackageVerificationCode** is RECOMMENDED: recommended by “NTIA SBOM Minimum elements” | (7.21) External Reference: e.g. ExternalRef: PACKAGE-MANAGER purl pkg:pypi/django@1.11.1 |  |  |

### **6.4.2 References**

<https://github.com/OpenChain-Project/Telco-WG/blob/main/OpenChain-Telco-SBOM-Guide_EN.md#32-spdx-elements-to-be-included-in-an-openchain-telco-sbom-guide-compatible-document>

# **Appendices**

## **Appendix.1 SBOM Samples**

Figure.1 presents an overview of the entire software supply chain. Based on this supply chain framework, we examine sample SBOM Documents that conform to various legislation and guidelines, illustrating how such documents are exchanged between entities. Each sample is designed to reflect the specific requirements and mapping criteria as detailed in the corresponding chapters.

In Figure4, an illustrative scenario is provided where the entity distributing App-A, App-A Provider is aware that App-A depends on Lib-C. However, it is noted that critical details, such as the version number and Unique Identifier of Lib-C, can not be specified.

This annotation highlights the potential gaps in information that can occur during the SBOM Document exchange process and underscores the importance of precise component identification for ensuring both security assurance and license compliance. These gaps represent the "Known Unknowns" that must be addressed to achieve full compliance and mitigate risks.

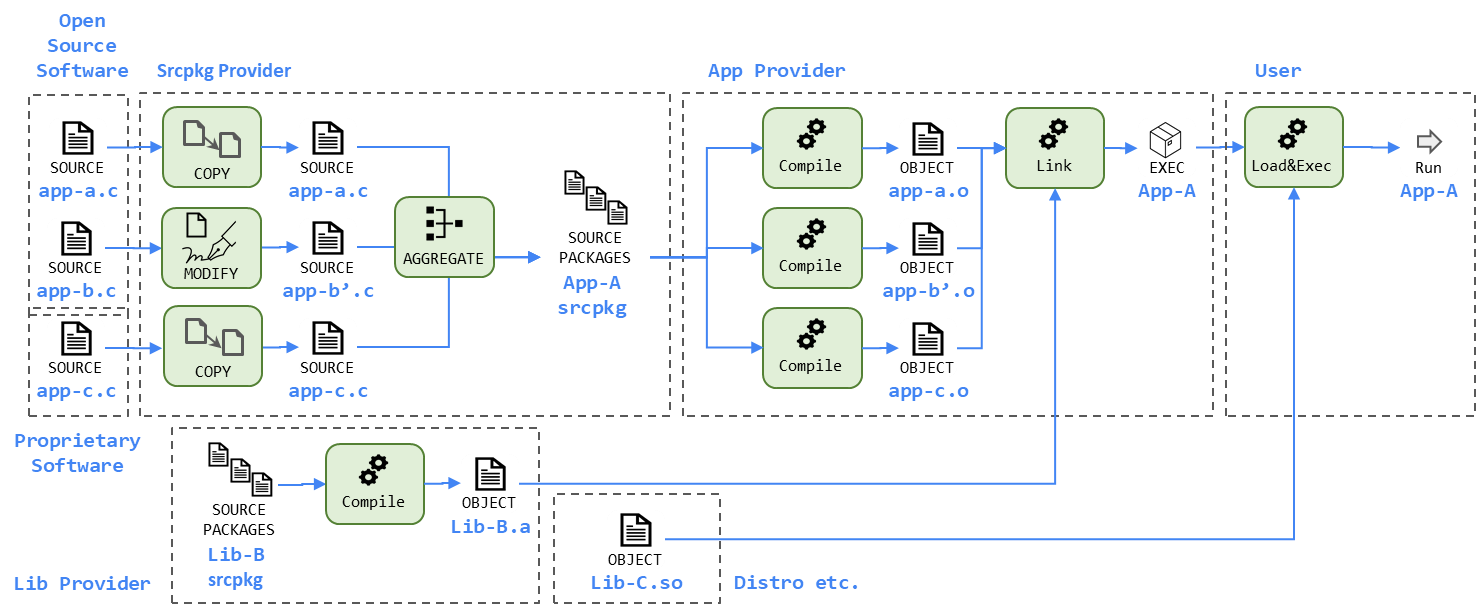


Figure.1

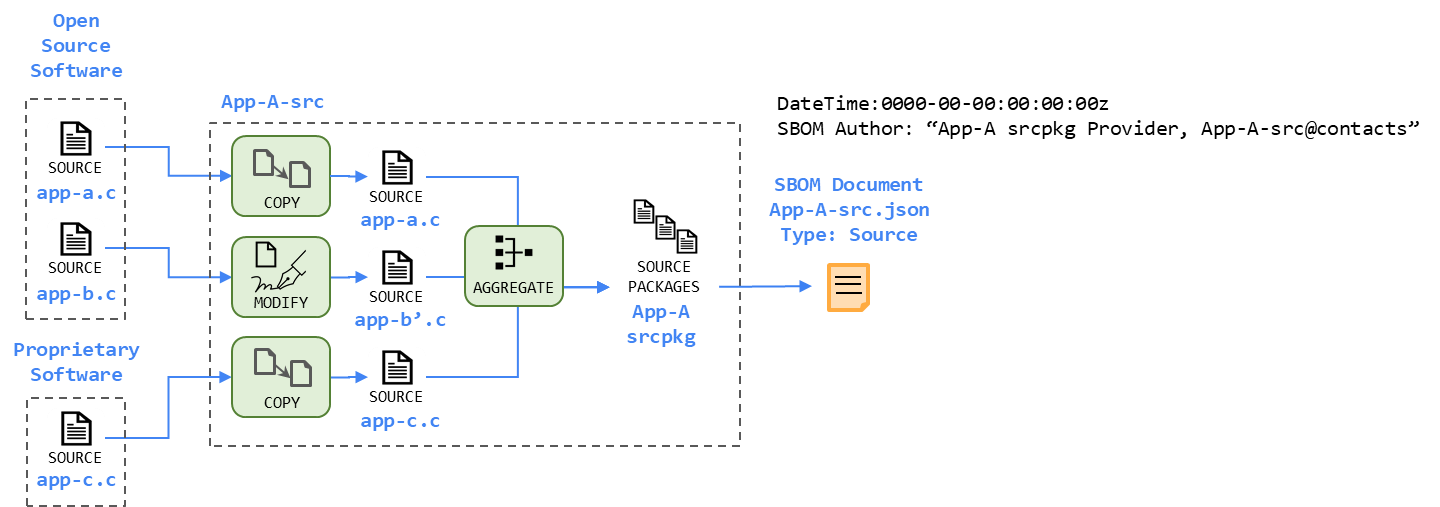


Figure.2

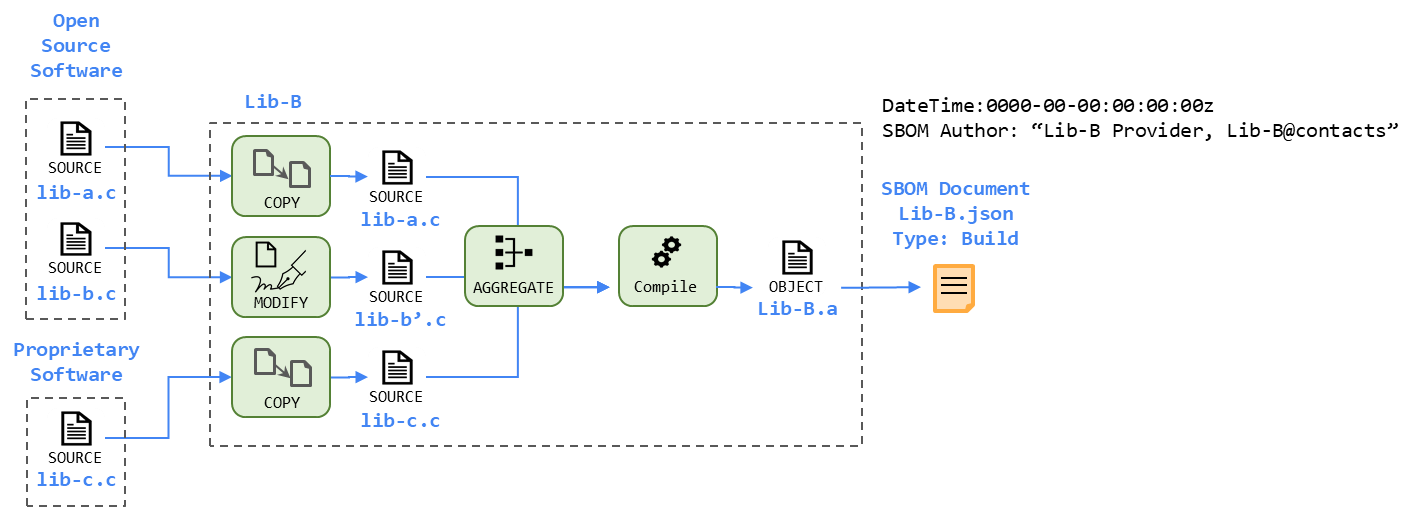


Figure.3

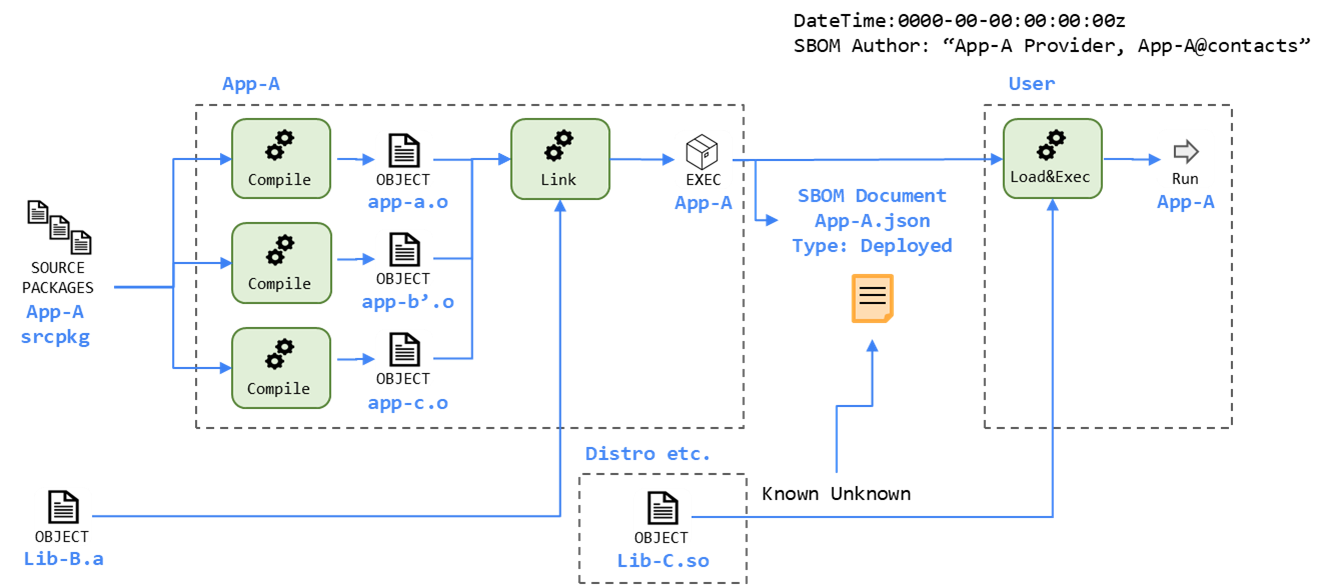


Figure.4

## **Appendix.2 Authors**

Signed-off-by:

## **Appendix.3 Version History**

Two penguins holding rings with text

AI-generated content may be incorrect.

This guide is one of many documents created, shared and maintained by the OpenChain Project to support a more trusted supply chain.

**Our vision** is a supply chain where open source is delivered with trusted and  
consistent process management information.

**Our mission** is to make that happen.

**We Maintain Standards:**

OpenChain ISO/IEC 5230  
The international standard for open source license compliance programs

OpenChain ISO/IEC 18974  
The industry standard for open source security assurance programs

And we maintain over 1,000 documents of supportive reference material, ranging from policy templates to self-certification checklists to training guides.

Learn more and get everything for free at www.openchainproject.org