2026 Minimum Elements for a Software Bill of Materials (SBOM)

Publication: July 29, 2026

Guidance at a Glance

Title	2026 Minimum Elements for a Software Bill of Materials (SBOM)
Original Publication	July 29, 2026
Executive Summary	The U.S. Cybersecurity and Infrastructure Security Agency (CISA), in partnership with the co-authoring organizations, updated the Minimum Elements for a Software Bill of Materials (SBOM) to reflect current SBOM needs, while preserving the core principles of the document published in 2021 by the National Telecommunications and Information Administration (NTIA). Increased adoption and implementation from stakeholders across the software ecosystem has uncovered new use cases and applications for SBOM data. SBOM tooling, in response to the increased adoption and implementation of SBOM, has advanced far beyond the capability and functionality of tools available in 2021. These advancements enable organizations requesting SBOMs to demand more information about their software components and supply chain.
Key Actions	▪ Organizations should request SBOMs that satisfy the updated SBOM Minimum Elements. ▪ Organizations should use available tools to generate, ingest, and analyze SBOM data. ▪ Organizations should generate SBOMs that satisfy the updated SBOM Minimum Elements.
Intended Audience	Organizations that produce software, procure software, or operate software.

Version History

Version	Date	Revision Description
1.0	July 12, 2021	Initial version.
2.0	August 22, 2025	Published draft document with updates that reflect advancements in SBOM tooling and technological implementation since 2021.
2.1	July 29, 2026	In collaboration with co-authoring organizations, updated document based on comments received in response to a request for comment on a draft of the SBOM Minimum Elements.

Table of Contents

Introduction

A software bill of materials (SBOM) has emerged as a key building block in software security and software supply chain risk management. An SBOM is a nested inventory, a list of ingredients that make up software applications and systems. Organizations that produce, procure, and operate software can use SBOM data to better understand their software supply chain. Increased software supply chain visibility can drive risk management decisions, including addressing known and newly discovered vulnerabilities and risks.

This document, authored by the U.S. Cybersecurity and Infrastructure Security Agency (CISA) and the following partners (hereafter referred to as the "authoring organizations"), updates and replaces the Minimum Elements for a Software Bill of Materials published by the National Telecommunications and Information Administration (NTIA). The elements outlined in this document set forth the minimum expected data fields and the practices and processes for an SBOM.

- U.S. National Security Agency (NSA)
- U.S. Federal Bureau of Investigation (FBI)
- Australian Signals Directorate's Australian Cyber Security Centre (ASD's ACSC)
- Canadian Centre for Cyber Security (Cyber Centre)
- Czech National Cyber and Information Security Agency (NÚKIB)
- French Cybersecurity Agency (ANSSI)
- Germany's Federal Office for Information Security (BSI)
- Indian Computer Emergency Response Team (CERT-In)
- Italy's National Cybersecurity Agency (ACN)
- Japan's Ministry of Economy, Trade and Industry (METI)
- Japan's National Cybersecurity Office (NCO)
- Republic of Korea's National Intelligence Service/National Cyber Security Center (NIS/NCSC)
- Korea Internet and Security Agency (KISA)
- Netherlands' National Cyber Security Centre (NCSC-NL)
- New Zealand's National Cyber Security Centre (NCSC-NZ)
- Poland's Research and Academic Computer Network (NASK)
- Slovakia's National Security Authority (NBU)

Since NTIA published The Minimum Elements For a Software Bill of Materials (hereafter, SBOM Minimum Elements) in 2021, organizations, experts, and practitioners across the software ecosystem have integrated SBOMs into their software development and security practices. The role of SBOMs in software supply chain transparency is recognized internationally:

- International partners participating in the Secure by Design initiative recognize SBOM as a key secure product development practice.
- Eighteen international cybersecurity organizations outline the benefits of SBOMs in A Shared Vision of Software Bill of Materials (SBOM) for Cybersecurity.

- Japan's Ministry of Economics, Trade, and Industry provides SBOM implementation guidance for software suppliers in Guidance on Introduction of Software Bill of Materials (SBOM) for Software Management.
- The European Union's Cyber Resilience Act (Regulation (EU) 2024/2847) requires manufacturers of products with digital elements to provide an SBOM as part of the technical documentation.
- The German Federal Office for Information Security describes technical requirements for SBOMs in BSI TR-03183-2: Cyber Resilience Requirements for Manufacturers and Products - Part 2: Software Bill of Materials (SBOM).
- India's CERT-In outlines requirements for SBOM and other bills of materials in Technical Guidelines on Bill of Materials.

SBOM tooling has advanced, driven by the growing number of organizations generating, sharing, consuming, and analyzing SBOMs. These advancements enable organizations requesting SBOMs to demand more information about their supply chain and software components than they could have in 2021.

The minimum elements discussed in this document apply to all software. Some types of software, such as artificial intelligence (AI) software systems[1] and software as a service (SaaS) in cloud environments, may require additional elements. For different types of software, additional data fields, practices, and processes may be beneficial. Any effort to improve software transparency, regardless of software type, should begin with the application of the SBOM minimum elements.

Notable Updates

This 2026 update to the SBOM Minimum Elements reflects current SBOM needs while preserving the core principles of the 2021 version. For example, automation remains critical for driving security at scale. The 2026 update incorporates public feedback received in response to the Request for Comment on 2025 Minimum Elements for a Software Bill of Materials. The revisions improve data quality and support a broader range of SBOM use cases. See **Appendix A: Table of Minimum Elements Data Fields** for the SBOM Minimum Elements and definitions. See **Appendix B: Summary of Minimum Elements Changes From 2021** for a detailed summary of changes from the 2021 version.

New SBOM elements (to support more risk-informed decisions about software security) include:

- SBOM Author Signature
- SBOM Data Format Name
- SBOM Data Format Version
- SBOM Generation Context
- SBOM Tool Name
- SBOM Tool Version
- SBOM Version
- Component Hash Value
- Component Hash Algorithm
- Component License

Major SBOM updates (to clarify scope and specify expectations) include:

- SBOM Author
- Component Identifiers
- Component Producer
- Component Version

[1] CISA and the Group of Seven (G7) international partners—Germany, Canada, France, Italy, Japan, the United Kingdom, and the European Union—released joint guidance, *Software Bill of Materials for AI – Minimum Elements*, in May 2026.

- Accommodation of Updates to SBOM Data
- Coverage
- Explicitly Identifying Unknown Information
- Machine-Processable Data

Minor SBOM updates (to improve information quality and align with technological developments) include:

- SBOM Timestamp
- Component Name
- Component Dependency Relationship
- Distribution and Delivery
- Frequency

Note: This update removes the Access Controls element and incorporates access control considerations in Distribution and Delivery.

Why SBOM: Enhancing Transparency and Security

SBOMs are a critical step toward achieving software component transparency, illuminating the software supply chain, and better positioning organizations to make risk-informed decisions regarding the components that make up their software systems. Components can include traditional software, firmware, AI systems, and SaaS. An SBOM serves as an ingredients list for software, providing organizations with data about the makeup of their software. Organizations can transform that data into insights that can drive machine-speed actions to reduce risks to the security of their software systems.

An SBOM is roughly hierarchical in structure. Software consists of components, each of which may have subcomponents. Each subcomponent may also be composed of subcomponents, and so on. Components (and subcomponents) may come from a "first party" or "third party." First-party components come from the party that is authoring the SBOM and are identifiable as independent, trackable units of software. Third-party components are from an external source. Each component should have an SBOM that captures its subcomponents and the hierarchy of its dependencies.

Effective mechanisms for sharing and using software data must be machine-processable and scalable. SBOMs capture component data in a machine-processable format, supporting analysis, sharing, and management. Organizations can map SBOM data to other sources, such as security advisories or organization-level approved/not approved software databases, to improve other priority practices such as secure software development or vulnerability management. SBOMs will not resolve all software security and supply chain concerns, but they are a necessary step for enabling and empowering risk-informed security decision-making.

Why Minimum Elements: Driving Security at Scale

The minimum elements specify the baseline an SBOM should meet. The volume of software used by modern organizations is too high for manual processes to effectively assess and mitigate risks. Given the role of software in enabling essential functions and critical infrastructure, software that no one tracks and manages poses a significant cybersecurity threat, including to critical infrastructure and government

systems.[2] Shared expectations for SBOMs help organizations apply SBOM data to software and supply chain security practices and facilitate information sharing.

Organizations across the software ecosystem and practitioners around the world have recognized the valuable role of SBOMs in increasing software and supply chain transparency and have contributed to the growth of SBOM adoption and advancement of SBOM technological implementation. As SBOM adoption continues to spread globally across industries and sectors, the importance of harmonizing expectations for SBOM will only increase.

Scope

The minimum elements outlined in this document apply to SBOMs for all software, including open source software, AI software, and SaaS. While additional elements may be necessary to make more complex software systems (such as AI or SaaS) transparent, an SBOM should still include the minimum elements. Additional elements that may be necessary to enable transparency for specific types of software are out of scope for this document.

The minimum elements do not create new requirements; they refine how organizations should generate and request SBOMs. Data management, storage practices, and precise encoding details are out of scope.

SBOM Minimum Elements

The SBOM minimum elements support an evolving approach to software transparency by capturing both the technology and the functional operation. Over time, the capabilities for transparency in the software supply chain will improve, with future efforts incorporating more detail and technical advances. **Appendix B: Summary of Minimum Elements Changes From 2021** provides the change log for updates relative to the 2021 SBOM Minimum Elements. The minimum elements set baseline expectations for:

- **Data Fields:** The data that makes up the SBOM document.
- **Practices and Processes:** How an entity engages with and documents the SBOM data.

Each **Data Fields** element need not correlate directly with a particular data field in an SBOM data format, and an implemented data field may satisfy one or more of the minimum elements. For example, a data field for a component-name/component-version pair may satisfy the elements of Component Name and Component Version. SBOM data formats may also use different field names.

The **Practices and Processes** elements outline principles that guide SBOM operations across the software lifecycle. SBOM implementation—how the data is generated, shared, analyzed, and otherwise used—differs widely, in part due to the variety of SBOM use cases.[3] For example, an SBOM may identify software subcomponents by enumerating the requisite data about each subcomponent within one large SBOM

[2] Cybersecurity and Infrastructure Security Agency, "BOD 26-02: Mitigating the Risk from End-of-Support Edge Devices," February 5, 2026, https://www.cisa.gov/news-events/directives/bod-26-02-mitigating-risk-end-support-edge-devices.

[3] For examples, see OWASP Foundation, Authoritative Guide to SBOM, October 21, 2025, https://cyclonedx.org/guides/OWASP_CycloneDX-Authoritative-Guide-to-SBOM-en.pdf and Open Source Security Foundation, Improving Risk Management Decisions with SBOM Data, September 18, 2025, https://openssf.org/wp-content/uploads/2025/09/Improving_Risk_Management_Decisions_with_SBOM_Data.pdf.

document or by linking to separate SBOMs for each subcomponent. Either approach can satisfy the baseline software supply chain visibility expectations defined in the Coverage element.

Data Fields

The core of an SBOM is a consistent, uniform structure that captures and presents information used to understand the components that make up software. Data field elements describe baseline information about the target component (the component for which the SBOM is generated) and all subcomponents enumerated in the SBOM document. The goal of these elements is to enable sufficient identification of a component and its subcomponents. Identification supports organizational risk management practices for tracking the software supply chain and correlation with other data sources, such as vulnerability, change management, or license databases. Some data fields may present similar data in different contexts across the many components and subcomponents, but SBOM authors should err on the side of duplication or redundant information for the sake of clarity and identification.

There are two categories of data field elements: SBOM Metadata and Component Data. SBOM Metadata elements capture information about the SBOM document itself. Component Data elements capture information about the target component and all enumerated subcomponents.

SBOM Metadata

SBOM Metadata elements capture information about the SBOM document itself. The data supports SBOM management; SBOM analysis tools leverage these data fields to improve usability of the SBOM data.

SBOM Author (Major Update)

Definition: The name of the entity that creates the SBOM data for the target component.

The SBOM Author element contains a string that identifies the entity that generated the SBOM for the target component. This element captures the entity operating the tool to generate the SBOM, not the tool itself. Inputs for this field should use full names and should not use acronyms (unless the official name of the entity that generated the SBOM includes an acronym).

The SBOM Author element is distinct from the Component Producer element, which identifies the entity that created the target component. The SBOM Author and the Component Producer field contents may be identical if the same entity created the target component and generated the SBOM. In cases where an entity that did not create the target component generates the SBOM for the target component, the field contents will be different.

SBOM Author Signature (New)

Definition: A digital signature attributable to the SBOM author.

The SBOM Author Signature element provides assurance that the claimed signatory signed the information and that the information was not modified after signature generation.[4] Digital signatures should use a signature algorithm approved for secure use, according to regulations or recommendations from relevant

[4] U.S. National Institute of Standards and Technology (NIST), "Digital Signatures," https://csrc.nist.gov/Projects/Digital-Signatures.

authorities, such as the U.S. National Institute of Standards and Technology (NIST) Digital Signature Standard (DSS), the International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 14888-4:2024, or the European Union Agency for Cybersecurity (ENISA) Agreed Cryptographic Mechanisms.

SBOM Data Format Name (New)

Definition: The name of the data format used to represent the SBOM data.

The SBOM Data Format Name element identifies the data format used to generate and consume the SBOM. See **Machine-Processable Data** for information on SBOM data formats.

SBOM Data Format Version (New)

Definition: Identifier designated by the SBOM data format to specify the version of the data format.

The SBOM Data Format Version element identifies the version of the data format indicated in the SBOM Data Format Name element. Versions declared to be deprecated by the organizations maintaining the data format should not be used.

SBOM Generation Context (New)

Definition: The relative software lifecycle phase and data available at the time the SBOM author generated the SBOM.

The SBOM Generation Context element conveys information about the component data documented in the SBOM. Component data may differ based on the phase of the software lifecycle during which the SBOM was generated. General software lifecycle references such as "before build," "build," and "after build," as well as more specific identifiers, can satisfy this element.[5] For example, an SBOM generated from source code could be identified as "before build," and binary analysis tools can generate an SBOM "after build."

SBOM Timestamp (Minor Update)

Definition: Record of the date and time of the most recent update to the SBOM data.

The SBOM Timestamp element identifies when the SBOM author last changed the SBOM data, either manually or using a software tool. Each version of an SBOM will have a new timestamp. The content of this field should adhere to RFC 9557.[6]

SBOM Tool Name (New)

Definition: The name of the tool used by the SBOM author to generate or amend the SBOM.

The SBOM Tool Name element contains a string that identifies the software tool that the SBOM author used to generate the SBOM. Inputs for this field should use full names and should not use acronyms (unless the tool's official name includes an acronym).

---

[5] "Build" refers to compilation, interpretation, or another process that makes source code executable.

[6] Ujjwal Sharma and Carsten Bormann, "Date and Time on the Internet: Timestamps with Additional Information," RFC 9557, April 2024, https://www.rfc-editor.org/info/rfc9557.

SBOM Tool Version (New)

Definition: Identifier for the version of the tool identified in the SBOM Tool Name element.

The SBOM Tool Version element captures the tool version and allows an organization to identify a specific code delivery. If no version identifier is available, the SBOM author should indicate that the information is unknown.

SBOM Version (New)

Definition: Identifier designated by the SBOM author to specify a change in the SBOM document from a previously identified version or to indicate that it is the first version.

The SBOM Version element indicates a relationship with earlier iterations of an SBOM and signals that the SBOM author made changes to the previous iteration. The SBOM version is the version of an SBOM for each component-name/component-version pair.

The SBOM Version element may use Semantic Versioning;[7] if the SBOM author uses Semantic Versioning, the major version of a published SBOM following these minimum elements should be "1."[8] SBOM authors should use minor version and patch version to indicate changes to the content of the SBOM elements as appropriate. If using an identifier such as a serial number to differentiate between versions of an SBOM, use of the identifier should conform to relevant standards (for example, RFC 9562 for unique identifiers).[9] SBOM authors should update the version for an SBOM for a given component-name/component-version pair when editing data about the target component.

Component Data

Component Data elements capture information about the target component (the component for which the SBOM is generated) and all enumerated subcomponents. Analysis of this data provides information about the components and their relationships and enables better informed risk management decisions regarding the target component.

Component Producer (Major Update)

Definition: The name of an entity that creates, defines, and identifies components.

The Component Producer element contains a human-readable string that identifies the entity that created or developed the target component.[10] The Component Producer element can enable the identification of a point of contact for software security concerns. Only one organization should be identified as the component producer for a given component. This practice will help distinguish between components of the

---

[7] Tom Preston-Werner, "Semantic Versioning 2.0.0," https://semver.org/spec/v2.0.0.html.

[8] The next minor version update would be 1.1, and the next major version update would be 2.

[9] K. Davis, B. Peabody, and P. Leach, "Universally Unique IDentifiers (UUIDs)," RFC 9562, May 2024, https://www.rfc-editor.org/info/rfc9562.

[10] The term "producer" should not be confused with the term of art in the European Union (EU) for software "manufacturer." While all manufactures are producers, not all producers are manufacturers.

For more information, see: EU, "Regulation (EU) 2024/2847 of the European Parliament and of the Council, Article 3," Official Journal of the European Union, October 23, 2024, https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=OJ:L_202402847.

same name produced by different component producers. Inputs for this field should use full names and should not use acronyms (unless the component producer's official name includes an acronym).

For open source software projects, the Component Producer field remains important. Naming practices may vary depending on how the project develops, distributes, and maintains packages and files. Certain ecosystems provide conventions for naming component producers through package managers. Otherwise, the SBOM author should use the original project or maintaining organization, when available. If there is no clear indication of component producer, the SBOM author should explicitly indicate that the component is of unknown provenance to acknowledge the lack of traceability.

The Component Producer element replaces the Supplier Name element from the 2021 SBOM Minimum Elements. Supplier Name has proven ambiguous in practice, particularly around distributors of software. Although reduced by the change to "Component Producer," some ambiguity will remain until there is a consensus methodology for identifying entities that create software.

Component Dependency Relationship (Minor Update)

Definition: The relationship between two components, where one component is necessary for the operation of the other.

The Component Dependency Relationship element reflects that a component includes another component. This information specifies the relationship between the target component and one of its subcomponents and supports the capability to build a dependency graph.[11] SBOM data formats may structure information about these subcomponents as SBOM elements embedded in the SBOM. Alternatively, an SBOM document can link to separate SBOM documents for each of the target component's dependencies.

Component Hash Value (New)

Definition: The output generated from applying a cryptographic hash algorithm to an executable component artifact.

The Component Hash Value element contains an American Standard Code for Information Interchange (ASCII), hexadecimal encoded value that is the result of providing the executable component artifact as input to a cryptographic hash algorithm.[12] If the SBOM author does not have access to the executable component artifact, then the SBOM author should indicate the value is unknown.

Component Hash Algorithm (New)

Definition: The cryptographic algorithm used to compute the Component Hash Value of the software component.

---

[11] The expected practice when a developer uses another project is to make an internal fork, name it, possibly edit it, and use an include or import function for the local fork. When a developer follows less sustainable practices, such as copying and pasting functional code from another project, copied code is effectively a dependency that is better tracked as a fork and a dependency relationship. An SBOM Author may document the borrowed code as an inclusion dependency by naming it as if it were a local fork. Tracking borrowed code in this way improves vulnerability management.

[12] If the format allows alternative formatting and encodings, those options should be provided in a way that is machine-processable and automatable.

The Component Hash Algorithm element documents the algorithm that produced the Component Hash Value to allow validation of the integrity of the target component. The SBOM author should identify the algorithm using Internet Assigned Numbers Authority (IANA) Hash Function Textual Names. The algorithm should be approved by a relevant authority, such as NIST.[13]

Component Identifiers (Major Update)

Definition: Identifiers used to identify a component or serve as a look-up key for relevant databases.

The Component Identifiers element contains at least one software identifier associated with the target component.[14] Machine-processable, unique software identifiers support automated analysis. This field should use common software identifiers, such as Common Platform Enumeration (CPE)[15] and Package-URL (PURL).[16] This field may also include universally unique identifiers (UUID), organization-specific identifiers, commit hashes, and intrinsic identifiers such as OmniBOR[17] and Software Hash Identifier (SWHID).[18] If there are multiple software identifiers, the SBOM author should include all of them.

Component License (New)

Definition: The identifier(s) for the license(s) under which the software component is available.

The Component License element captures the identifiers for the licenses under which the software component is available. The license identifiers should allow an organization receiving the SBOM data to find the full details of the licenses. When possible, the SBOM author should convey this information in a machine-processable fashion, such as by using System Package Data eXchange (SPDX) license identifiers.[19] If no identifier is available, the contents should indicate through other means where the full license details are available, such as via a URL. This field should also include information about the existence of proprietary license conditions. If the SBOM author is not aware of the license information, then the SBOM author should indicate that license information is unknown.

Component Name (Minor Update)

Definition: The name assigned by the component producer to a software component.

The Component Name element identifies the software component in a human-readable way. The component producer determines the component name. This field is distinct from the Component Identifiers field. Data formats implementing the Component Name element should allow for multiple entries to capture alternate names. Inputs for this field should use full names and should not use acronyms (unless the component's official name includes an acronym).

---

[13] For NIST's approved algorithms, see NIST's Hash Functions.

[14] U.S. CISA, "Software Identification Ecosystem Option Analysis," October 26, 2023, https://www.cisa.gov/resources-tools/resources/software-identification-ecosystem-option-analysis.

[15] U.S. NIST, "CPE: Common Platform Enumeration," https://nvd.nist.gov/products/cpe.

[16] Ecma International, "ECMA-427: Package-URL (PURL) Specification," December 2025, https://ecma-international.org/publications-and-standards/standards/ecma-427/.

[17] OmniBOR, "OmniBOR Specification v0.1," https://omnibor.io/spec/v0.1/.

[18] SWHID, "The SWHID Specification Version 1.2," https://www.swhid.org/specification/v1.2/; ISO, "ISO/IEC 18670:2025," April 2025, https://www.iso.org/standard/89985.html.

[19] SPDX, "SPDX License List," https://spdx.org/licenses/.

Component Version (Major Update)

Definition: Identifier used by the component producer to specify a change in a software component from a previously identified version or to indicate that it is the first version.

The Component Version element captures the target component's version and allows an organization to identify a specific code delivery. If the component producer does not provide a version, then the SBOM author should indicate that the information is unknown.

Practices and Processes

An SBOM is more than a structured set of data; how an entity engages with and documents the SBOM data—its SBOM practices and processes—is as important as the data itself. An organization should explicitly address these elements in any policy, contract, or arrangement to ask for or provide SBOMs. Some of these elements, such as Frequency, are straightforward. Others, such as Access, involve multiple practices that build on both existing approaches and novel concepts.

Accommodation of Updates to SBOM Data (Major Update)

Organizations should accommodate updates to SBOM data, including corrections. SBOM authors should correct errors promptly. Organizations may consider errors, whether stemming from SBOM author practices or selection of inadequate tools, in organizational risk management decisions.

Coverage (Major Update)

An SBOM should include information for all components that make up the target software, including transitive dependencies. There is no minimum depth. If there are multiple instances of a software component with differences in metadata, each instance should be listed separately with the appropriate dependency relationship. An SBOM may exclude non-code files; however, it may include security-relevant files such as configuration files. SBOMs should provide a comprehensive listing of the components to facilitate recipients' risk-based decisions. For example, from a vulnerability management perspective, the recipient of an SBOM should be able to conclude that a newly reported vulnerability does not affect them if the SBOM does not list the component associated with the vulnerability. Linking to other SBOM documents—for example, an SBOM generated for a subcomponent or dependency—can support SBOM authors in meeting the requirements for the Coverage element, provided that the recipient has access to all linked SBOMs.

Distribution and Delivery (Minor Update)

SBOMs should be available promptly to those who need them. Access controls may limit the sharing of SBOM data with unauthorized parties but should not prevent information sharing between authorized parties or restrict organizations from integrating SBOM data into trusted security tools. There are multiple ways of sharing SBOM data. For example, an SBOM can accompany installation. Alternatively, an SBOM can be accessible through a version-specific URL, an application programming interface (API) to a database, or a public repository. Any such software service or offering should operate in accordance with the provider's security policy.

Explicitly Identifying Unknown Information (Major Update)

If information required for any of the data fields is not provided, the SBOM author should explicitly state whether the information is unknown to the SBOM author or whether the SBOM author is withholding the

information from the SBOM. The explicit identification of information that is unknown can provide SBOM recipients with useful information about both the target software and the SBOM author. The SBOM author should have a process for recipients to ask about any redacted, security-related information. Organizations may consider an SBOM incomplete if the SBOM author withholds essential component data.

Frequency (Minor Update)

Each software version or update should have an associated SBOM. When a component producer issues a new build or release, they (or the SBOM author) should also generate a new SBOM to reflect the changes. This includes software builds that integrate updated components or dependencies. When a component producer (or SBOM author) discovers new details about the underlying components or corrects an error in the existing SBOM data, they (or the SBOM author) should issue a revised SBOM.

Machine-Processable Data (Major Update)

Automation support is critical for managing software component data at scale, particularly across organizational boundaries. SBOM implementations should be compatible with each other to support automation due to the volume of data, diverse use cases, and variety of tools involved with SBOMs. The two data formats currently widely used by software ecosystem stakeholders to generate and consume SBOMs are SPDX[20] and CycloneDX.[21] These data formats are a product of open, international processes and are both machine-processable and human-readable.

Minimum support for automation means supporting all data formats that are widely used, open source, and compatible with existing data formats. Supported data formats should be reassessed regularly. If a format is broadly incompatible, no longer maintained, or ineffective for SBOM use cases, it should no longer be used. This approach accomplishes the goal of building on existing tools for ease of adoption while supporting future evolution and extensibility.

Component producers or SBOM authors may choose their preferred SBOM data format based on factors that may be specific to organizations, industries, or sectors. Organizations should accept any widely used, interoperable, and machine-processable SBOM format. However, organizations should avoid accepting SBOMs for new software generated in deprecated[22] versions of any format to maintain compatibility with SBOM consumption and management tools.

Discussion

As SBOMs and SBOM tooling continue to advance, four key areas warrant further consideration and potential guidance: cloud software (including SaaS), AI software, validation, and correlation with security advisories. Cloud and AI software may warrant additional SBOM elements. Validation and correlation with

---

[20] SPDX, "The System Package Data Exchange," https://spdx.dev/; ISO, "ISO/IEC 5962:2021," August 2021, https://www.iso.org/standard/81870.html.

[21] CycloneDX, "CycloneDX," https://cyclonedx.org/; Ecma International, "ECMA-424: CycloneDX Bill of Materials Specification," December 2025, https://ecma-international.org/publications-and-standards/standards/ecma-424/.

[22] Avoid deprecated versions or features, as these may be removed in the future or may only be kept for compatibility purposes. For more information, see MDN Web Docs, "Browser Compatibility Data Project," https://github.com/mdn/browser-compat-data/blob/main/schemas/compat-data-schema.md#status-information.

security advisories present both valuable opportunities and challenges for all SBOMs. These topics are areas of ongoing exploration and future work.

SBOM for Software as a Service in Cloud Environments

The shared responsibility between SaaS producers and SaaS operators for risk mitigation and the frequent changes to SaaS complicate applying the SBOM model from on-premises software to SaaS. Both SaaS producers and operators play a role in administering and securing the system. This shared responsibility may limit some SBOM use cases. The frequency of SBOM delivery for SaaS may also need further specification. The nature of cloud software and modern development practices results in high-frequency changes to the software. Delivering and receiving SBOMs at such high frequency may overwhelm both the SaaS producer and the SaaS operator. The usual precise specification of the software deployed to the container for SaaS systems may offset these challenges. Additionally, distribution mechanisms such as APIs can accommodate rapid code changes in continuous-integration and continuous-delivery (CI/CD) processes by capturing snapshots. Despite these challenges, SBOMs for cloud and SaaS use cases retain all the usual software transparency benefits. Further specifications, including potential additional elements for an SBOM for SaaS/cloud software, may warrant additional exploration.

SBOM for AI Software Systems

AI software systems, like cloud and SaaS environments, may have additional components that are not captured by the minimum elements. As AI becomes increasingly embedded in software systems and potentially plays a role in critical infrastructure, organizations should retain and improve visibility into the components of the system to appropriately manage associated risks. Since AI is software, the minimum elements illuminate many components that organizations should be aware of, but there may be additional useful supply chain data—for example, information such as what AI engineers commonly discuss under the titles of "model cards" or "data cards." There are ongoing discussions about SBOM for AI use cases and possible data fields that AI users can request.[23] However, this document does not introduce additional elements for SBOMs for AI systems.

Validation

SBOMs, like other security data, may warrant different levels of trust. An organization receiving an SBOM may be concerned about verifying the source of the SBOM data and confirming its integrity. The SBOM Author Signature element provides a place for SBOM authors to include information that provides assurance the recipient is receiving the SBOM as the SBOM author created it. Implementation of this element should use existing software signing infrastructure, tools, and key management.[24]

In addition to assurance that the SBOM received is the same as the SBOM created, organizations may seek to confirm the accuracy, coverage, and completeness of SBOM data. These process-based properties have

[23] For more information, see the G7 Cybersecurity Working Group's joint guidance Software Bill of Materials (SBOM) for Artificial Intelligence – Minimum Elements (authored by CISA and G7 international partners).

[24] For more information, see NIST Special Publication 800-57 Part 1 Revision 5: Recommendation for Key Management, Part 1 – General.

different quality measures and different assurance expectations than simply a signature and are out of the scope of the SBOM Minimum Elements. For these quality measures, organizations may use sources such as open source software repositories or binary analysis tools to detect components not listed in the SBOM.

Correlating SBOM With Security Advisories

Security advisories, such as Vulnerability Exploitability eXchange (VEX)[25] and the Common Security Advisory Framework (CSAF)[26] documents, provide additional security information about relevant software components. Security advisories communicate information about the vulnerability status of a software component. A component producer can use security advisories to notify its downstream users or customers that one of their software components is vulnerable and that they are working on a fix (or whatever vulnerability status is most appropriate). Vulnerability status updates not only highlight security issues but also help avoid unnecessary effort if a trusted source deems that potential risk does not translate to actual risk.

This security information is particularly impactful for vulnerability management. After learning new vulnerability information, an organization can assess whether its SBOMs contain or reference a vulnerable component, significantly improving response time compared to organizations without SBOMs. Security advisories such as VEX and CSAF can further improve the efficiency of vulnerability management processes by narrowing the scope of at-risk software components.

Conclusion

As new use cases emerge and technology evolves, SBOM minimum elements should evolve to continue to provide transparency into software components. An SBOM alone is data about software components. Analysis of SBOMs transforms data into insights about associated risks. Organizations can then use those insights to drive security decisions about their software systems. Vulnerability management tools that can ingest SBOMs, analyze the data, and map it to other data sources will enable organizations to leverage data, insights, and actions driven by SBOMs.

The authors play a key role in advancing and refining SBOM adoption and implementation, in collaboration with the SBOM community and stakeholders across the software ecosystem. The experiences and expertise of each of the members of the SBOM community have been invaluable in shedding light on SBOM adoption and implementation challenges, sharing lessons learned across industries, and refining technological aspects of SBOMs. The authoring organizations will continue to leverage their positions as leaders in cybersecurity to facilitate information sharing and bring together experts across the community to identify and promote solutions to software supply chain challenges.

Disclaimer

CISA and the authoring organizations do not endorse any commercial entity, product, company, or service, including any entities, products, or services linked within this document. Any reference to specific commercial

[25] OpenVEX, "OpenVEX Specification," https://github.com/openvex/spec/blob/main/OPENVEX-SPEC.md.
[26] OASIS, "Common Security Advisory Framework Documentation," https://oasis-open.github.io/csaf-documentation/.

entities, products, processes, or services by service mark, trademark, manufacturer, or otherwise, does not constitute or imply endorsement, recommendation, or favoring by CISA and the authoring organizations.

This publication, Minimum Elements for a Software Bill of Materials, is of a general nature and is not intended to, and does not constitute advice for compliance, regulatory, or legal purposes. This publication is subject to change and may not constitute the most up-to-date guidance or technical information. Readers of this publication should confer with their respective advisors and subject matter experts to obtain advice based on their individual circumstances.

Acknowledgements

CISA and the authoring organizations acknowledge all stakeholders who provided input and comments on the draft document.

The Directorate-General for Communications Networks, Content and Technology (DG CONNECT) of the European Commission contributed to this document.[27]

[27] This document does not interpret European Union law nor is it meant to be a guidance for implementation of Union law. The document does not bind the European Commission. The Directorate General for Communications Networks, Content and Technology (DG CONNECT) contributed to the drafting of the document in order to cooperate on and emphasize shared cybersecurity principles. However, as this document is a multilateral effort, not all of its elements reflect Union law. Entities falling within the scope of Union law might use this document for information purposes only.

References

CycloneDX. "CycloneDX." https://cyclonedx.org/.

Davis, K., B. Peabody, and P. Leach. "Universally Unique IDentifiers (UUIDs)." RFC 9562, May 2024. https://www.rfc-editor.org/info/rfc9562.

Ecma International. "ECMA-424: CycloneDX Bill of Materials Specification." December 2025. https://ecma-international.org/publications-and-standards/standards/ecma-424/.

Ecma International. "ECMA-427: Package-URL (PURL) Specification." December 2025. https://ecma-international.org/publications-and-standards/standards/ecma-427/.

EU. "Regulation (EU) 2024/2847 of the European Parliament and of the Council, Article 3." Official Journal of the European Union, October 23, 2024. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32024R2847&qid=1778510136341#tit_1.

IANA. "Hash Function Text Names." December 16, 2024. https://www.iana.org/assignments/hash-function-text-names/hash-function-text-names.xhtml.

ISO. "ISO/IEC 5962:2021." August 2021. https://www.iso.org/standard/81870.html.

ISO. "ISO/IEC 18670:2025." April 2025. https://www.iso.org/standard/89985.html.

MDN Web Docs, "Browser Compatibility Data Project." https://github.com/mdn/browser-compat-data/blob/main/schemas/compat-data-schema.md#status-information.

OASIS. "Common Security Advisory Framework Documentation." https://oasis-open.github.io/csaf-documentation/.

OmniBOR. "OmniBOR Specification v0.1." https://omnibor.io/spec/v0.1/.

OpenVEX. "OpenVEX Specification." https://github.com/openvex/spec/blob/main/OPENVEX-SPEC.md.

SPDX. "SPDX License List." https://spdx.org/licenses/.

SPDX. "The System Package Data Exchange." https://spdx.dev/.

SWHID. "The SWHID Specification Version 1.2." https://www.swhid.org/specification/v1.2/.

Tom Preston-Werner. "Semantic Versioning 2.0.0." https://semver.org/spec/v2.0.0.html.

Ujjwal Sharma and Carsten Bormann. "Date and Time on the Internet: Timestamps with Additional Information." RFC 9557, April 2024. https://www.rfc-editor.org/info/rfc9557.

U.S. CISA. "Software Identification Ecosystem Option Analysis." October 26, 2023. https://www.cisa.gov/resources-tools/resources/software-identification-ecosystem-option-analysis.

U.S. NIST. "CPE: Common Platform Enumeration." https://nvd.nist.gov/products/cpe.

U.S. NIST. "Digital Signatures." https://csrc.nist.gov/Projects/Digital-Signatures.

U.S. NIST. "Hash Functions." https://csrc.nist.gov/projects/hash-functions.

Appendix A: Table of Minimum Elements Data Fields

Please see **Table 1** for minimum elements data fields.

Table 1. Minimum Elements Data Fields

Data Field	Definition
Component Dependency Relationship	The relationship between two components, where one component is necessary for the operation of the other.
Component Hash Algorithm	The cryptographic algorithm used to compute the Component Hash Value of the software component.
Component Hash Value	The output generated from applying a cryptographic hash algorithm to an executable component artifact.
Component Identifiers	Identifiers used to identify a component or serve as a look-up key for relevant databases.
Component License	The identifiers for the licenses under which the software component is available.
Component Name	The name assigned by the component producer to a software component.
Component Producer	The name of an entity that creates, defines, and identifies components.
Component Version	Identifier used by the component producer to specify a change in a software component from a previously identified version or to indicate that it is the first version.
SBOM Author	The name of the entity that creates the SBOM data for the target component.
SBOM Author Signature	A digital signature attributable to the SBOM author.
SBOM Data Format Name	The name of the data format used to represent the SBOM data.
SBOM Data Format Version	Identifier designated by the SBOM data format to specify the version of the data format.
SBOM Generation Context	The relative software lifecycle phase and data available at the time the SBOM author generated the SBOM.
SBOM Timestamp	Record of the date and time of the most recent update to the SBOM data.
SBOM Tool Name	The name of the tool used by the SBOM author to generate or amend the SBOM.
SBOM Tool Version	Identifier for the version of the tool identified in the SBOM Tool Name element.
SBOM Version	Identifier designated by the SBOM author to specify a change in the SBOM document from a previously identified version or to indicate that it is the first version.

Appendix B: Summary of Minimum Elements Changes From 2021

The following summarizes the changes between this document and the 2021 NTIA SBOM Minimum Elements. Changes that significantly impact implementation are reported. An example of a reported change is if a field name (such as Supplier Name) has been updated (such as to Component Producer). An example of an unreported change is if another field's description has been updated to cross-reference the new field name. Most elements had an adjective, either "SBOM" or "Component," added to the element title for clarity; these changes are not individually reported below. Elements are listed in alphabetical order; elements included in the 2021 NTIA SBOM Minimum Elements are listed under the name as it appears in the 2021 NTIA SBOM Minimum Elements.

Access Control (Removed)

Changes: Remove Access Control element from the SBOM minimum elements.

Explanation: The development in SBOM practices since 2021 has made the Access Control element no longer necessary as a stand-alone requirement. The Distribution and Delivery element now includes all relevant access control considerations.

Accommodation of Mistakes (Major Update)

Changes: Replace Accommodation of Mistakes element with Accommodation of Updates to SBOM Data element. Refine the definition for what types of changes to SBOM data should be accommodated.

Explanation: The relative immaturity of SBOM adoption and implementation in 2021 made explicitly accommodating mistakes necessary. Technological developments since 2021 have significantly improved SBOM data quality, such that recipients can expect SBOM data to be accurate.

Author of SBOM Data (Major Update)

Changes: Replace Author of SBOM Data element with SBOM Author element.

Explanation: The term "SBOM Author" is common in SBOM-related discussions and more accurately captures the role of the entity generating the SBOM.

Automation Support (Major Update)

Changes: Remove Software Identification (SWID) Tags from list of data formats. Replace Automation Support element with Machine-Processable Data element. Move Machine-Processable Data element under Practices and Processes.

Explanation: SWID tags are not a widely used SBOM data format for which multiple tools exist. SBOM data formats and SBOM tools have prioritized machine processability and established using machine-processable data as a core practice for SBOM implementation.

Component Dependency Relationship (Minor Update)

Changes: Clarify that "dependency" refers to a component that is necessary to the operation of another component.

Explanation: The updated definition better scopes the element to capture the minimum information needed for an organization to assess and monitor the risks associated with a software component.

Component Hash Algorithm (New)

Changes: Add Component Hash Algorithm element.

Explanation: Identifying the algorithm used to generate the hash value is critical for making component hashes useful.

Component Hash Value (New)

Changes: Add Component Hash Value element.

Explanation: Hashes play a significant role in existing software and supply chain security processes.

Component License (New)

Changes: Add Component License element.

Explanation: A software component's license can aid organizations in managing the risk of copyright infringement claims, by facilitating the proper use of software according to the terms of its license. Organizations should consider a software component's license and re-license information in their risk management plans, because incorrect or fraudulent license information can affect an organization's ability to rely on the software or to obtain support from the producer.

Component Name (Minor Update)

Changes: Allow multiple entries.

Explanation: Being able to add multiple entries better enables organizations to map the data to the appropriate software component.

Depth (Major Update)

Changes: Replace Depth element with Coverage element. Define Coverage as including horizontal breadth (in addition to vertical breadth) of software component information.

Explanation: The Depth element was previously defined as capturing software component information for top-level dependencies only. This definition reflected the capabilities of SBOM tooling at the time rather than the depth of information needed to make informed security decisions. Rather than define the degree of depth for dependency information required, the Coverage element includes horizontal as well as vertical breadth of software component information. The advances in SBOM tooling since 2021 make specifying this breadth of software component information feasible.

Distribution and Delivery (Minor Update)

Changes: Simplify and clarify Distribution and Delivery element.

Explanation: Given advances in SBOM sharing practices, a more concise definition is appropriate.

Frequency (Minor Update)

Changes: Rewrite the element to use updated terminology.

Explanation: The update clarifies the Frequency element and reflects current terminology but leaves the intent of the element unchanged.

Known Unknowns (Major Update)

Changes: Replace Known Unknowns element with Explicitly Identifying Unknown Information element. Separate information that is known but purposefully withheld from that which is unknown to the SBOM author.

Explanation: As SBOM implementation has increased, organizations have raised concerns regarding the ambiguity of the Known Unknowns element. This document updates the name of the element to clarify that identifying unknown information about a component should be a practice implemented throughout the SBOM lifecycle rather than a data field included in a data format. Distinguishing between information that is known to the SBOM author and purposefully withheld from information that is unknown to the SBOM author provides further clarification for implementation of this element.

Other Identifiers (Major Update)

Changes: Replace Other Unique Identifiers element with Component Identifiers element. Specify that at least one common software identifier should be included and that this field may also include universally unique identifiers, organization-specific identifiers, commit hashes, and intrinsic identifiers.

Explanation: While there is not yet a universal solution to the problem of software identification, there has been progress. The changes reflect both the maturity of the discussion around software identification and the maturity of the software identifier formats themselves.

SBOM Author Signature (New)

Changes: Add SBOM Author Signature element.

Explanation: The SBOM author's digital signature provides assurance regarding the integrity and authenticity of the SBOM data.

SBOM Data Format Name (New)

Changes: Add SBOM Data Format Name element.

Explanation: Identifying the SBOM data format within the data schema can support SBOM tools in reading and analyzing the SBOM data.

SBOM Data Format Version (New)

Changes: Add SBOM Data Format Version element.

Explanation: The SBOM data format version can support SBOM tools in reading and analyzing the SBOM data.

SBOM Generation Context (New)

Changes: Add SBOM Generation Context element.

Explanation: Provide more information regarding the source of the data represented in the SBOM. SBOM data may vary depending on the lifecycle phase of the software during which the SBOM author generates the SBOM. This additional context better informs organizations for assessing software risk.

SBOM Tool Name (New)

Changes: Add SBOM Tool Name element.

Explanation: The addition of this element provides information regarding the manner in which the SBOM Author generates the SBOM.

SBOM Tool Version (New)

Changes: Add SBOM Tool Version element.

Explanation: The version of the SBOM tool used to generate the SBOM can be important for efficiently ingesting and analyzing SBOM data.

SBOM Version (New)

Changes: Add SBOM Version element.

Explanation: A given component may have multiple associated SBOMs, updated over time to improve the accuracy of the data. Version tracking will support organizations in managing SBOMs and monitoring risks.

Supplier Name (Major Update)

Changes: Replace Supplier Name element with Component Producer element.

Explanation: "Component Producer" better aligns with terms commonly used in SBOM discussions. The term further clarifies that the field refers to the entity that originated the software. Establishing a fallback designation for an unknown component producer (indicating unknown provenance) helps ensure transparency and highlights potential risks in the software supply chain.

Timestamp (Minor Update)

Changes: Clarify that entries should adhere to RFC 9557.

Explanation: Following RFC 9557 for entries better supports automation and interoperability.

Version of the Component (Major Update)

Changes: Replace Version of the Component element with Component Version element. Specify that if the component producer does not provide a version, then the SBOM author should indicate that the version is unknown.

Explanation: Being able to associate software component data with a specific version is important for processes such as vulnerability management. Indicating that a software component's version is unknown provides insight to an organization regarding the SBOM author's knowledge of the software's provenance.