SBOM FAQ

Table of Contents

GENERAL

Q: What is an SBOM?

A: A Software Bill of Materials (SBOM) is a complete, formally structured list of components, libraries, and modules that are required to build (i.e. compile and link) a given piece of software and the supply chain relationships between them. These components can be open source or proprietary, free or paid, and widely available or restricted access. For details, see Section 2 of "Framing Software Component Transparency: Establishing a Common Software Bill of Material (SBOM)": https://www.ntia.gov/files/ntia/publications/framingsbom_20191112.pdf

Q: Who uses an SBOM and for what?

A: Most SBOM usage fits under one or more of three perspectives: those who produce software, those who choose software, and those who operate software.

- For those who produce software, SBOMs are used to assist in the building and maintenance of their supplied software.
- For those who choose or purchase software, SBOMs are used to inform pre-purchase assurance, negotiate discounts, or plan implementation strategies.
- For those who operate software, SBOMs are used to inform vulnerability management, asset management, to manage licensing and compliance, and to quickly identify software or component dependencies and supply chain risks.

Q: Who should have an SBOM?

A: In today's world, software touches every part of our life and spans across industries, with much of it built on third-party code and open source software. Anyone who is concerned about better supporting their software products internally, supporting their customers, and positively differentiating themselves in the marketplace should consider creating SBOMs and providing them to support their customers. Over time, more SBOM requirements may emerge, such as the FDA mandate for medical device manufacturers. For additional information on use cases and benefits of SBOMs, see https://www.ntia.gov/files/ntia/publications/ntia_sbom_use_cases_roles_benefits-nov2019.pdf

Q: Who creates and maintains an SBOM?

A: An SBOM is created and maintained by the manufacturer or supplier of the software.

Q: When is an SBOM changed or maintained by a software supplier?

A: A new SBOM should be created for every new release of a component. Changes to components require corresponding changes to SBOMs to be valid. For details on when to create an SBOM, see Section 4.2 of “Framing Software Component Transparency: Establishing a Common Software Bill of Material (SBOM)”:
https://www.ntia.gov/files/ntia/publications/framingsbom_20191112.pdf

Q: How will SBOM data be shared?

A: SBOM data can be shared in a variety of ways including, but not limited to:

- Readme in the distribution kit, on disk next to the binary
- Manufacturer website
- Some centralized or trusted third party’s website
- Full content from device
- Pointer from device (MUD)
- Human-readable files provided to the purchaser

NTIA stakeholders continue to review how SBOM data can be shared effectively.

Q: How does an SBOM help in the event of a cyberattack?

A: When flaws or vulnerabilities are discovered in a given component, SBOMs are used to quickly identify software that is affected by the vulnerable component, to assess its usage, and to understand the risk introduced by the vulnerable component. The ability to identify vulnerabilities allows software suppliers to produce patches or provide other remediation options; allows consumers to apply mitigations independently of the software supplier; and allows the identification of software that is not affected. This enables focusing on the software that may be affected.

Q: Where can I find more information about the NTIA SBOM process? How do I get involved?

A: To learn more about the NTIA Multistakeholder process for SBOM – including scope, definitions, tools, formats, community-drafted documents, and existing state of practices – visit: www.ntia.gov/sbom, www.ntia.gov/softwareTransparency, or reach out to Allan Friedman at afriedman@ntia.gov.

Concerns about SBOMs

Q: Won't SBOMs be a "roadmap to the attacker"?

A: Theoretically, yes. In reality, the defensive benefits of transparency far outweigh this common concern as SBOMs serve more as a "roadmap for the defender". All information is dual-edged, but insufficient software transparency affords attackers asymmetrical advantages. First, attackers don't need SBOMs; mass, indiscriminate attacks like WannaCry serve to remind us that foreknowledge is not a prerequisite to cause harm. Second, attackers and their tools can more easily identify software components; conversely, it is often quite challenging, disruptive, inefficient, and even unlawful for defenders to determine the same. Third, attackers of any single product can already find *human-readable* target components – licensing requirements have been increasingly requiring disclosure for decades. SBOMs seek to level the playing field for defenders by providing additional transparency – at enterprise scale – with standard, *machine-readable* decision support.

Q: Is this intellectual property or source code disclosure?

A: No. Your proprietary source code remains yours to share or to keep confidential at your discretion. Concerns about exposing the internals of your code's operation are likewise unfounded, as these software components are just "a piece of the puzzle", not anything close to the "whole completed puzzle" that represents your program.

Q: Will SBOMs increase my licensing costs or licensing commitments?

A: No. The awareness gained by creating an SBOM allows the manufacturer to address unknown licensing commitments that may be lurking in your programs. This permits the manufacturer to address these issues, either through licensing fees or securing more favorable licensing terms, thus avoiding fines, lawsuits, and licensing commitments such as exposure of your proprietary code.

Details of SBOM execution

Q: What should be included in an SBOM?

A: An SBOM should contain some combination of the following baseline information: author name, supplier name, component name, version string, component hash, unique identifier, and relationship. Licensing, pedigree, provenance, should also be included, if available. For detailed information about SBOM baseline component information, see section 2.2 of "Framing Software Component Transparency: Establishing a Common Software Bill of Material (SBOM)":
https://www.ntia.gov/files/ntia/publications/framingsbom_20191112.pdf

Q: Some software components are made up of other software components themselves. Can an SBOM show that hierarchy?

A: Yes. SBOMs can provide hierarchical information. Each component that is included in an SBOM should have an SBOM of its own if it also includes components. The SBOMs supplied with each software component taken together represent all levels of the hierarchy required to sufficiently understand the software and its various parts. Such an SBOM is analogous to a manufactured product *multilevel bill of materials.*
https://en.wikipedia.org/wiki/Bill_of_materials#Multi-level_BOM

Q: How deep in the dependency graph should an SBOM enumerate?

A: It depends on the intended audience and the medium of communication. In the case of a machine-readable SBOM, the minimum viable model is one layer deep with the goal of recursing up the supply chain. The FDA would like to see it as complete as possible, but they understand that complete SBOMs will take time.

Role specific

Q: How can SBOMS be leveraged as a Purchaser?

Having an SBOM informs and enables the following, prior to purchasing decisions:

- Catalog various parts of the software and their inter-relationships
- Understand chain of licensing of the software product
- Understand complexity of the software (dates, versions of various parts of the software)