Carnegie Mellon University
Software Engineering Institute

SOFTWARE BILL OF MATERIALS FRAMEWORK: LEVERAGING SBOMS FOR RISK REDUCTION

Charles M. Wallen, Christopher Alberts, Michael Bandor, & Carol Woody

June 2023

Introduction

Managing evolving technology and the cybersecurity risks that come with today's computing environments is a foundational aspect of doing business. Software-related risk is becoming a leading consideration for mitigating risks because its role in computing continues to grow. Systems are increasingly reliant on software that is replacing hardware, especially in systems of systems environments. Code has become a commodity; software relies on a combination of programming content and shared code to accomplish targeted objectives. Recent events, such as SolarWinds® and Log4j™, demonstrate the scale of cybersecurity disruption that can result from a lack of vigilance in managing the components that combine to form software that meets the budget and schedule demands of global markets [Liu 2021].

New methods and perspectives must be used to strengthen cybersecurity and establish more resilient environments. The dependencies that can pose risks to systems require more innovative and collaborative solutions. Software is a prime example; those managing software must consider its dependencies and risks in new ways. Teams of business and technology experts must collaborate and develop new techniques for identifying potential risks and proactively managing (i.e., tracking, analyzing, and mitigating) risks. Software bills of materials (SBOMs) can help facilitate the building of those new techniques and foster the necessary collaboration. The U.S. Department of Commerce (DOC) defines an SBOM as follows [DOC 2021]:

> ***An SBOM*** *is a formal record containing the details and supply chain relationships of various components used in building software. In addition to establishing these minimum elements, this report defines the scope of how to think about minimum elements, describes SBOM use cases for greater transparency in the software supply chain, and lays out options for future evolution.*

SBOM-driven techniques have grown in popularity as necessary and effective resources for gathering information about the components and sources/suppliers of content that typically comprise software systems. Early efforts to innovate SBOM methods focused on defining data elements and managing known vulnerabilities. Several emerging information and risk management methods identify critical data and connect support teams, suppliers, and stakeholders to reduce risk.

Risks and Challenges

Software development often lacks effective resistance to attack and controls for preventing tampering by malicious actors. Typically, software development project resources are insufficiently integrated and can be affected by cost, schedule, and compliance constraints. These characteristics leave operational environments with weaknesses and vulnerabilities that must be addressed later in the system's lifecycle. A lack of cost-schedule-risk balance can lead to shortcuts and software that does not conform to secure coding and secure software development practices. There is a pressing need to implement more transparent and predictable mechanisms for ensuring that software functions securely and as intended.

No software is free of risks. Defects exist even in the highest quality software. For example, best-in-class code can have up to 600 **defects** per million lines of code (MLOC), while average quality code has around 6,000 defects per MLOC. Some of these defects are weaknesses that can lead to vulnerabilities. Research indicates that up to 5% of software defects are security vulnerabilities [Woody 2014]. As a result, best-in-class code can have up to 30 **vulnerabilities** per MLOC. For average quality code, the number of security vulnerabilities can be as high as 300 vulnerabilities per MLOC. Reducing the number of security vulnerabilities in code is an important part of software development. Using secure coding practices, peer reviews, and code analysis tools are important ways to identify and correct known weaknesses and vulnerabilities in code.

Third-party risk is a major challenge for organizations seeking to limit their exposure to cybersecurity risks from general outsourcing and supplier-provided software that is critical to system functions. There is often little transparency into the components, sources, and suppliers involved in a system's development and ongoing operation. SBOM information can help reduce the uncertainties that result from that lack of transparency. An essential aspect of addressing supplier risk is being able to access information about supplier inputs and their relative importance, and then managing mitigations to reduce risk.

The lack of integration among a system's technology teams, including suppliers, is another source of risk where SBOM information can help reduce risk and improve efficiency. Not only do teams often work in stovepipes, but the teams who use supplier software and technology services/products may also neglect to engage or oversee those suppliers. Development and support teams often work independently with varying objectives and priorities driven by cost and schedule demands that do not fully consider (existing or potential) risk.

Poor team communication and collaboration often leads to inefficient and disconnected outcomes. SBOMs and the information they provide can help teams support a system and its software. They can serve as a source and "rally point" for establishing more unified risk and resilience objectives. To achieve these benefits, SBOMs require innovations that target managing the information they provide and establishing more standardized practices and process.

The Role of SBOMs and Opportunities for Their Use

SBOMs have come to the forefront of efforts to strengthen cybersecurity risk management tools, which was a highlight of Executive Order (EO) 14028, *Improving the Nation's Cybersecurity*, issued on May 12, 2021 [White House 2021]. EO 14028 requires U.S. government agencies to enhance software supply chain security and integrity, with a priority on addressing critical software.[1] A key aspect of enhancing software supply chain security and integrity is transparency. Implementing SBOMs for critical software will help establish transparency in the software supply chain. Therefore, EO 14028 calls for standards, procedures, and criteria for providing SBOMs for products directly or publishing them on a public website.

As a result of EO 14028, the National Telecommunications and Information Administration (NTIA) and U.S. Department of Commerce (DOC) published *The Minimum Elements for a Software Bill of Materials (SBOM)* [DOC 2021]. It includes a list of SBOM elements, use cases, and options for the future direction of SBOMs. In this document, the message is clear: "[…] these are the initial steps and requirements needed to support the basic use cases. There is more work to be done to expand transparency in the software supply chain and to support visibility for securing software" [DOC 2021].

Our survey of SBOM publications and guidance revealed a strong emphasis on defining the content and format of SBOMs. Establishing a standard for SBOM content is an important aspect of the problem; however, organizations also need guidance on how to plan for, develop, deploy, and use SBOMs. As a result, our team at the Software Engineering Institute (SEI) focused our research activities on the SBOM lifecycle.[2] SBOMs must support more, including (1) proactively considering how to best manage risks posed by third parties and (2) developing effective mitigations as new or emerging threats and vulnerabilities emerge. There are simply too many moving parts and risks in today's software-driven technology environments to simply rely on ad hoc or poorly organized SBOM practices and processes.

There is broad support for increasing the utility of SBOMs. A necessary and beneficial next step is to develop leading practices and supporting processes. Developing more comprehensive and collaborative SBOM practice frameworks will offer techniques for effectively establishing and managing proactive software information and risk management programs. Using SBOMs can also provide software developers, integrators, and risk managers with a unique opportunity to collect information that they can analyze, monitor, and act on to manage software components, suppliers/dependencies, provenance, vulnerabilities, and more—the possibilities are extensive.

We also recognize that the SBOM lifecycle does not exist in isolation. Rather, it is performed in an organizational context. In addition to the core lifecycle activities, we must consider enabling and supporting other activities, such as those performed by program management, organizational support

---

[1] *Critical software* is defined as software that performs functions critical to achieving trust, such as affording or requiring elevated system privileges or direct access to networking and computing resources.

[2] The SBOM lifecycle refers to the set of activities required to plan for, develop, and use an SBOM.

(e.g., information technology, risk management, change management), and third parties. Going forward, it will be important to look creatively at how SBOM data can be used to manage software risk and efficiency and provide support to teams that can benefit from collaborative efforts to solve problems.

Leveraging Our Research to Strengthen SBOM Use

To address the need for more comprehensive, SBOM-informed risk management, our SEI team used principles and concepts from their work on the Acquisition Security Framework (ASF) to draft a set of SBOM practices and processes [Alberts 2022]. ASF is a framework of leading practices and processes for managing the security/resilience of applications and systems of systems. The ASF facilitates an integrated approach to cyber risk management across a system's lifecycle and its supply chains. The motivation for developing the ASF came from a need for innovative methods to manage many of the same challenges that SBOMs address (e.g., third-party/acquisition risk, growing role of software in systems, lack of integration among support teams, complexity of systems). Key to meeting these challenges is taking a multi-disciplinary approach to managing cyber risk over a system's lifecycle.

ASF principles and concepts are built on a legacy foundation that includes the following:

- software engineering concepts from Capability Maturity Model Integration℠ (CMMI)
- risk management concepts from Operationally Critical Threat, Asset, and Vulnerability Evaluation® (OCTAVE)
- resilience management methodologies from the CERT® Resiliency Management Model (CERT-RMM)

These methodologies resulted in highly influential bodies of knowledge that have informed the subsequent development of tools and techniques that have been further refined through extensive use across a range of sectors and industries. The organizational outcomes that result from the ASF approach are characterized by efficient and predictable environments and more manageable delivery and risk outcomes.

The ASF is designed to help a program coordinate managing engineering and supply chain risks across system components, including hardware, network interfaces, software interfaces, and mission capabilities. The ASF includes 51 goals and 334 practices spread across the following six practice areas:

1. Program Management
2. Engineering Lifecycle
3. Supplier Dependency Management
4. Support
5. Assessment and Compliance
6. Process Management

ASF goals and practices provide a roadmap for managing security and resilience across a system's lifecycle and supply chain. The ASF defines engineering-driven, risk-based practices and processes for building, deploying, and operating secure and resilient systems. Its principles and concepts also provide a foundation that can be applied to a variety of related problems, such as integrating SBOMs into an acquisition program's risk management practices.

Building the SBOM Framework

We started developing the SBOM Framework by reviewing published use cases. Based on this review, we developed core SBOM practices, which primarily focused on developing SBOMs (i.e., building and construction practices) and using them to manage known security vulnerabilities and associated risks (i.e., operational use practices).[3] We then expanded on this initial set of practices by considering a lifecycle perspective. Here, we identified practices for specifying requirements, developing plans, and allocating resources needed to build and use SBOMs. Finally, we identified practices for activities that enable and support operational use of SBOM data. These practices include management and support practices, third-party practices, and infrastructure practices. The result is an SBOM Framework comprising the following six goals:[4]

1. Requirements
2. Planning
3. Build/Construct
4. Deploy/Use
5. Manage/Support
6. Infrastructure

The framework includes 44 practices distributed across the six goals.[5] The framework provides a starting point for integrating SBOMs with a program's security risk management practices. As we collect lessons learned from piloting the framework and feedback from the community, we will update the framework's goals and practices as appropriate.

As our team developed the SBOM Framework, we leveraged opportunities to discuss its approach and design with organizations that are in the process of developing and expanding their SBOM

---

[3] An SBOM has multiple operational uses, including managing known security vulnerabilities, software versions and licenses, code reuse, and software end-of-life issues. The SBOM Framework focuses on managing security vulnerabilities and risks.

[4] There is not a separate goal for third-party practices in the SBOM Framework. Third-party practices are included in Goal 1 (Requirements) and Goal 5 (Manage/Support).

[5] See the appendix for a list of the SBOM Framework's goals and practices.

capabilities. The feedback from those discussions provided valuable input and validation of the approach we took with the SBOM Framework.

Leveraging SBOM Information

SBOMs have been primarily designed to help organizations build more structure into the management of software risks. That management must include not only identifying, but also effectively mitigating security and resilience risks in systems. Clearly, much more can be done to facilitate a broader benefit of using SBOMs. Information/data from SBOMs, while a key factor in managing risk, has many other possible uses and innovations.

Achieving effective SBOM results requires planning, tooling (because the scale is too great), resources trained to do the job, and measurement/monitoring. Information that can be gathered from an SBOM can offer insights into the challenges faced by the groups engaged in managing a system. Figure 1 depicts some of the support teams that could use and benefit from SBOM information and processes to improve software and systems. The SBOM Framework provided in this paper largely focuses on the risks posed by software components and the suppliers who develop/manage that software. There are many other potential uses of and innovation opportunities from SBOM use, particularly given the vast data that SBOMs can provide.

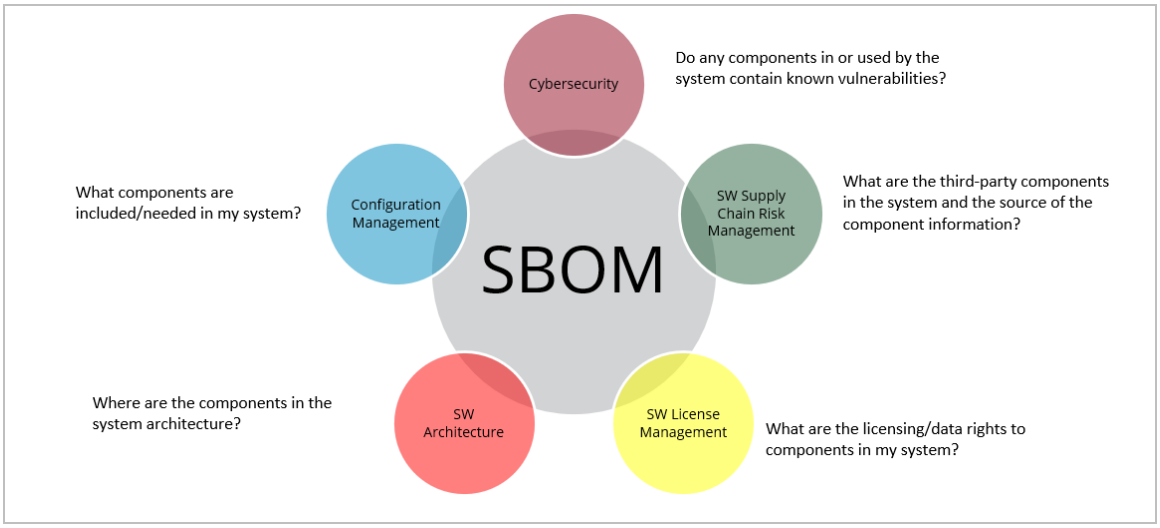

Figure 1: SBOM Relationships with Other Areas

Data about software risks and vulnerabilities is rich and extensive. Unfortunately, the risk information that SBOMs contain only adds to what is already an overwhelming conglomeration of content and potential information. Organizing and prioritizing that information is a challenge, and we hope that the SBOM Framework can help direct its users' efforts to establish the most effective approach for them.

Key to that approach will be the collaborative use of methods and tools by multiple teams involved in software and systems management. (See Figure 1.)

SBOM data analysis can help visualize difficult or, in some cases, unseen relationships and dependencies. These relationships and dependencies can be invaluable to teams who manage software in ever more complex technical environments. That benefit was described in *The Minimum Elements for a Software Bill of Materials (SBOM)* [DOC 2021]:

> *An SBOM should contain all primary (top level) components, with all their transitive dependencies listed. At a minimum, all top-level dependencies must be listed with enough detail to seek out the transitive dependencies recursively.*
>
> *Going further into the graph will provide more information. As organizations begin SBOM, depth beyond the primary components may not be easily available due to existing requirements with subcomponent suppliers. Eventual adoption of SBOM processes will enable access to additional depth through deeper levels of transparency at the subcomponent level.*

Conclusion and Next Steps

SBOMs are becoming crucial in managing software and systems risk and resilience. There are multiple efforts underway to expand the use of SBOMs. One driving factor is the reference to SBOMs in EO 14028. More importantly, there is wide and growing recognition that the risks posed by a lack of transparency in software must be addressed to help ensure security and promote resilience in systems. The practices and processes outlined in this SBOM Framework can provide a starting point to build that structure for SBOM efforts. The SBOM Framework addresses the establishment of processes to manage multiple SBOMs and the vast data that they can provide; however, those processes will likely require further tuning as pilot-related activities provide input about improvements and tooling.

This SBOM Framework can help promote the use of SBOMs and establish a more comprehensive set of practices and processes that organizations can leverage as they build their programs. Meanwhile, we will continue communicating broadly about the benefits and potential uses of SBOMs and gather feedback from pilots. We will continue to explore pilot opportunities. Where adoption of the SBOM Framework has occurred, we will study the lessons learned to help us in making refinements.

We plan to establish a more complete process and goal-oriented approach to managing software risk. The work presented in this paper is a start and will require further tuning and use. As stated in *The Minimum Elements for a Software Bill of Materials (SBOM)* [DOC 2021], there remains much to be done to improve software risk management transparency.

Appendix: SBOM Framework Goals and Practices

There are 44 practice questions across 6 groups of activities/processes that represent an overall program for managing software risks and vulnerabilities. The intent of this framework is to codify SBOM-related activities in a way that promotes identifying and mitigating risks to software posed by threats from weak code, third parties, and malicious actors.

The SBOM Framework uses a structure of topic-oriented goals and supporting questions to inquire about existing practices that support meeting those goals. Below are the SBOM Framework goals and practices, which are largely organized in the order they would be conducted.

SBOM Practice Goals

- Requirements
- Planning
- Build/Construct
- Deploy/Use
- Manage/Support
- Infrastructure

Requirements

Goal 1—SBOM requirements for the program are identified and managed.

The purpose of this goal is to ensure that SBOMs are integrated with the program's security/resilience activities.

1. Are program goals (e.g., reducing risk, managing system security/resilience) established for using SBOMs?
2. Are program requirements (e.g., required and desired data elements) established for SBOM content?
3. Are program requirements established for using SBOMs to support risk reduction and security/resilience activities?
4. Are criteria/triggers in place for reviewing SBOM requirements?
5. Are SBOM requirements updated periodically based on reviews and lessons learned?
6. Are baseline (i.e., boilerplate) SBOM requirements that apply to all program and system suppliers identified and documented?
7. Are criteria used to evaluate each supplier's ability to meet the program's SBOM requirements?
8. Are SBOM requirements included in formal agreements?

Planning

Goal 2—A plan for developing and using SBOMs is developed.

The purpose of this goal is to ensure that the programs have a plan for using SBOMs to manage software security/resilience risks.

1. Are standards, guidelines, and policies for implementing SBOM practices and artifacts established?
2. Are requirements established for implementing SBOM practices and artifacts to support risk management across the program or system?
3. Is sufficient funding allocated for implementing SBOM practices and artifacts across the program or system?
4. Are staff members assigned to implement SBOM practices and artifacts across the program or system?
5. Are roles and responsibilities established for SBOM practices?
6. Do stakeholders understand their roles in implementing, managing, and supporting SBOM practices?
7. Is SBOM training for technical and program staff members provided as needed?
8. Is a plan developed to manage SBOM practices and artifacts across the program or system?
9. Is the SBOM plan monitored and adjusted as needed?

Build/Construct

Goal 3—SBOM data is created for the system, subsystems, and components.

The purpose of this goal is to ensure that accurate and complete SBOM data is created and validated for the system, subsystems, and components.

1. Does the program's SBOM format meet specified requirements?
2. Is architecture information that identifies software components for each system and subsystem available?
3. Are information sources (e.g., engineering data, licensing data, results of software composition analysis) for creating an SBOM specified and used?
4. Are SBOMs for the system's commercial off-the-shelf (COTS) software, government off-the-shelf (GOTS) software, and open-source software (OSS) available?
5. Is an SBOM created or identified for each software component?
6. Are multiple SBOMs integrated to construct dependency trees for the system?
7. Is SBOM data validated for completeness and accuracy?

Deploy/Use

Goal 4—Vulnerabilities are identified and managed in SBOM software components, leading to reduced system risk.

The purpose of this goal is to ensure that SBOMs are used to manage vulnerabilities in the system's software components.

1. Are known vulnerabilities and available updates monitored for software components identified in the system's SBOM?
2. Are vulnerabilities in SBOM components identified?
3. Is the mission risk of each SBOM component assessed?
4. Are software updates prioritized based on their potential impact to mission risk?
5. Are software component reviews/updates conducted based on their mission-risk priorities?
6. Are vulnerability management status, risks, and priorities tracked for each software component?

Manage/Support

Goal 5—SBOM risks are managed for system components.

The purpose of this goal is to ensure that accurate, complete, and timely SBOM data is available for system components to effectively manage risk.

1. Are the suppliers for system components identified?
2. Is supplier data reviewed periodically and updated as needed?
3. Are SBOMs for system components identified, analyzed, and tracked?
4. Are SBOMs managed to ensure they are current?
5. Are the risks related to incomplete or missing SBOM data identified and mitigated?
6. Are risks and limitations related to managing and redistributing SBOM information identified and managed?
7. Is the provenance of SBOM data established and maintained?

Infrastructure

Goal 6—SBOM practices, software, and tools are selected, implemented, and managed.

The purpose of this goal is to ensure that SBOM practices, software, and tools are integrated into the program's infrastructure.

1. Are technical requirements for the SBOM infrastructure developed and documented?
2. Are SBOM practices, software, and tools selected and implemented?
3. Are SBOM practices, software, and tools monitored and managed?
4. Is the security/resilience of SBOM practices, software, and tools managed?
5. Are the integrity and authenticity of SBOM data validated and managed?
6. Is each SBOM and its related artifacts managed across the organization?
7. Is each SBOM and its related artifacts managed for each system?

References

[Alberts 2022]
Alberts, Christopher; Bandor, Michael; Wallen, Charles M.; & Woody, Carol. *Acquisition Security Framework (ASF): Managing Systems Cybersecurity Risk*. CMU/SEI-2022-TN-003. Software Engineering Institute, Carnegie Mellon University. 2022. https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=889215

[DOC 2021]
United States Department of Commerce (DOC). *The Minimum Elements For a Software Bill of Materials (SBOM).* DOC. 2021. https://www.ntia.doc.gov/files/ntia/publications/sbom_minimum_elements_report.pdf

[Liu 2021]
Liu, Nancy. SolarWinds to Log4j: More Risk Management Wake-Up Calls. *SDxCentral website.* December 21, 2021. https://www.sdxcentral.com/articles/news/solarwinds-to-log4j-more-risk-management-wake-up-calls/2021/12/

[White House 2021]
The White House. *Executive Order on Improving the Nation's Cybersecurity*. EO 10428. The White House. 2021. https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/

[Woody 2014]
Woody, Carol; Ellison, Robert; & Nichols, William. *Predicting Software Assurance Using Quality and Reliability Measures*. CMU/SEI-2014-TN-026. Software Engineering Institute, Carnegie Mellon University. 2014. https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=428589

Legal Markings

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM23-0161

Contact Us

Software Engineering Institute
4500 Fifth Avenue, Pittsburgh, PA 15213-2612

Phone: 412/268.5800 | 888.201.4479
Web: www.sei.cmu.edu
Email: info@sei.cmu.edu