Ke, Jessica - Intern

From: Robert A Martin <ramartin@mitre.org>
Sent: Tuesday, June 15, 2021 9:14 AM
To: SBOM_RFC; Friedman, Allan
Cc: Charles Clancy
Subject: MITRE reply to the NTIA Request for Comments on SBOMs

Below is MITRE's reply to the NTIA Request for Comments on SBOMs [Docket No. 210527-0117] RIN 0660-XC051.

Thank you for this opportunity to offer our thoughts on these important matters that are shaping the way forward in this critical area for supply chain security of software-based systems. This response is approved for Public Release; Distribution Unlimited. Case No: 21-01357-4.

Bob

--
Robert (Bob) Martin
Sr. Software and Supply Chain Assurance Principal Eng.
Cross Cutting Solutions and Innovation Dept
Cyber Solutions Innovation Center
MITRE Labs
MITRE Corporation
781-271-3001o
781-424-4095c

---

1. *Are the elements described above, including data fields, operational considerations, and support for automation, sufficient? What other elements should be considered and why?*

   MITRE believes that the set of data fields, operational considerations, and support for automation proposed by NTIA are good but missing a few considerations that are most appropriately captured in a SBOM when software is being created and will make automation and adoption more efficient and effective. Specifically:

   i. Add a date/time field to capture the creation date/time of the SBOM data and we suggest that an industry standard for that type of information be leveraged.
   ii. Add a data license field to inform recipients of the SBOM what the basic set rights or constraints the SBOM author has put on the SBOM itself.
   iii. Expand the list of components in a BOM to explicitly include the run-time dependencies like dynamic libraries the software utilizes, and the service calls it invokes. This will allow for a more complete understanding of what 3rd party capabilities the software makes use of in performing its functionality and thus are sources of possible risk to the enterprise running that software.
   iv. Add the privileges that the software requires to perform its intended functionality. This kind of information is commonly known for software in mobile devices but not well captured or conveyed in other environments but is key to monitoring a running application for aberrations from its expected behavior.
   v. Revise the dependency relationship data field to support a more flexible set of relationships between components, supporting use cases such as those above in iii and iv, (adding "invoked," "dynamically linked," and "required privilege," to the implied "included" relationship) and evolve it to cover non-dependency types of relationships.

vi. Add as a recommended set of fields for informative or analytic description information for the components, such as description, comment, or annotation fields.

Additionally, MITRE believes that the hash called out as a minimum SBOM field needs clarification and elaboration. Specifically:

i. Make the hash of the operational software that the SBOM is for required when possible so that the relationship between the SBOM and the software it is describing is clear and verifiable.
ii. Add, as a recommended field, the hash for each 3rd party component included in the software being described by the SBOM.
iii. Add, as an optional field, the hash of the source code directory used to build the software described in the SBOM using available standardized mechanisms for deterministically hashing directories of source files.

2. *Are there additional use cases that can further inform the elements of SBOM?*

Please see the answer to item 3.b. below for a use scenario for SBOM of services.

Additionally, MITRE believes there is a great potential for SBOM content to play a role outside traditional supply chain security in its interaction with other cybersecurity domain activities such as security operations (in addition to traditional IT management), cyber threat intelligence, cyber investigations, etc. where SBOMs may serve as potential whitelists for interpreting observed operational realities. This could be considered a different dimension to 3.d Integrity and Authenticity except instead of only being limited to the SBOM exchange scenario it could also apply to verifying integrity and authenticity of cyber observable content (files, etc.) observed on systems and networks by using the same SBOM integrity mechanisms.

3. *SBOM creation and use touches on a number of related areas in IT management, cybersecurity, and public policy. We seek comment on how these issues described below should be considered in defining SBOM elements today and in the future.*

a. *Software Identity: There is no single namespace to easily identify and name every software component. The challenge is not the lack of standards, but multiple standards and practices in different communities.*

MITRE believes that the current work on this within the NTIA's consensus building work on Software Component Transparency is adequately addressing this item.

b. *Software-as-a-Service and online services: While current, cloud-based software has the advantage of more modern tool chains, the use cases for SBOM may be different for software that is not running on customer premises or maintained by the customer.*

MITRE believes that there would be great utility to capturing, for audit and later forensics use, the SBOM of the Software-as-a-Service and online services when they are being invoked.

Rather than adding this as a requirement for all services MITRE proposes that this would be a new form of "Assured Service" offering that service providers could offer and that when customers chose to utilize the service, they would have the SBOM of the stack of the software providing the service as an additional aspect of that service that they could capture and log.

This information could then be used by the customer of the service to analyze new public vulnerability disclosures for any past exposures to possible exploitation and allow for a more

informed forensics analysis of network and other enterprise resources around the timeframe that they used the vulnerable software implementing the services they invoked.

This scenario was one of the nine captured in the white paper "Standardizing SBOM within the SW Development Tooling Ecosystem," 09-29-2019 from the Tool-to-Tool Software Bill of Materials Exchange joint working group of CISQ and the Object Management Group (OMG) https://drive.google.com/file/d/1H-o2sLNsj5Az2tyRiHRY3tXle7-2Yj1u/view?usp=sharing.

c. *Legacy and binary-only software: Older software often has greater risks, especially if it is not maintained. In some cases, the source may not even be obtainable, with only the object code available for SBOM generation.*

MITRE believes that while the preferred time and place to create an SBOM for a piece of software is at the point of building the software, there are many existant software items that could be usefully captured with SBOMs using different means. Flexibility in SBOM creation, such as by 3rd parties, will be a useful source of information about existing software and these same techniques and capabilities can offer a cross-checking capability to verify and validate SBOMs be analyzing the software described in an SBOM and validating key elements of the SBOM description.

d. *Integrity and authenticity: An SBOM consumer may be concerned about verifying the source of the SBOM data and confirming that it was not tampered with. Some existing measures for integrity and authenticity of both software and metadata can be leveraged.*

MITRE believes that the work of the Integrity Working Group of the Linux Foundation's SPDX project, which started as the Integrity Working Group of the Tool-to-Tool Software Bill of Materials Exchange joint working group of CISQ and the OMG until that group and SPDX merged efforts, has useful guidance on this topic. The current draft is available https://docs.google.com/document/d/1XZQD82l5Oq_Yr9g1t5vwAZe5UHMCZpHQhJ3L6v6zMVI/edit?usp=sharing.

e. *Threat model: While many anticipated use cases may rely on the SBOM as an authoritative reference when evaluating external information (such as vulnerability reports), other use cases may rely on the SBOM as a foundation in detecting more sophisticated supply chain attacks. These attacks could include compromising the integrity of not only the systems used to build the software component, but also the systems used to create the SBOM or even the SBOM itself. How can SBOM position itself to support the detection of internal compromise? How can these more advanced data collection and management efforts best be integrated into the basic SBOM structure? What further costs and complexities would this impose?*

MITRE believes that the current work on this within the Linux Foundation's in-toto project is better focused on this software supply chain integrity issue however, having SBOMs signed by their author and the software components signed by their author can enhance the ability to notice and act when there are discrepancies between the provenance of the delivered software and SBOM are different than the expected provenance of those items.

f. *High assurance use cases: Some SBOM use cases require additional data about aspects of the software development and build environment, including those aspects that are enumerated in Executive Order 14028. How can SBOM data be integrated with this additional data in a modular fashion?*

MITRE believes that information relevant to the high assurance use cases that is available when the software is built should be included as an optional part of the SBOM as soon as practical. By

including these as part of the SBOM the integrity of this information and its relationship with the SBOM can be addressed by signing the SBOM. The specific additional items we propose are:

i. The provenance information for the actions performed by the tool chain used to build the software. Potential items would be the actions, the tools (the name, versions, configuration, and environment), the performers, and the time details of the actions, as well as expanding the relationships the types of relationships between components.
ii. The signature(s) for the author of software described by the SBOM and the 3rd party software components included in that software, when the signature information is available.

g. *Delivery. As noted above, multiple mechanisms exist to aid in SBOM discovery, as well as to enable access to SBOMs. Further mechanisms and standards may be needed, yet too many options may impose higher costs on either SBOM producers or consumers.*

MITRE believes that the current work on this within the NTIA's consensus building work on Software Component Transparency is adequately addressing this item.

h. *Depth. As noted above, while ideal SBOMs have the complete graph of the assembled software, not every software producer will be able or ready to share the entire graph.*

MITRE believes that the current work on this within the NTIA's consensus building work on Software Component Transparency is on the right path to addressing this item, however, it is important to explore how the effort in SBOMs can utilize a consistent graph-based standard representation for the SBOM content. When this is done there is the possibility to merge/compose graphs from multiple sources to potentially mitigate some of the risk of incomplete depth SBOMs from a given provider. In other words, even if one producer creates an SBOM that only goes down to the level of a given component and no deeper, SBOM details for that component may be available through other vectors either directly from the producer of that component or another producer whose SBOM contains that component with details at a greater depth than the initial producer.

i. *Vulnerabilities. Many of the use cases around SBOMs focus on known vulnerabilities. Some build on this by including vulnerability data in the SBOM itself. Others note that the existence and status of vulnerabilities can change over time, and there is no general guarantee or signal about whether the SBOM data is up-to-date relative to all relevant and applicable vulnerability data sources.*

MITRE believes that the addition of the data/time to the SBOM can help alleviate the problem posed by including time-variant information in some SBOM formats since, with the date/time of the statement available, the inclusion of the vulnerability information becomes a statement about what was known at specific point in time. MITRE suggests that any asserted correlations between components within an SBOM and known vulnerabilities should wherever possible be references to the details of the vulnerabilities rather than attempting to duplicate the details within the SBOM. In other words, let the SBOM focus on asserting the correlations and not take on responsibility for publishing vulnerability details in an accurate and timely manner.

This separation of efforts assumes that the vulnerability reporting and recording community, like NVD/CVE, will include sufficient details to allow correlation between publicly known vulnerabilities and the SBOMs for the software with those vulnerabilities.

MITRE proposes that:

i. The inclusion of vulnerability information should be avoided for use cases other than those focused on capturing what was known about vulnerabilities at the point of building the software and its SBOM.
ii. SBOMs created after the building of the software should not include vulnerability information as SBOM data, rather this would be an attestation about the previous built software at the point in time of the analysis or testing that created this additional data about vulnerabilities and other types of analysis of the software.
iii. Vulnerability repositories and reporting efforts should be encouraged to include sufficient information to allow for the easy correlation between SBOMs and publicly reported vulnerabilities.

j. Risk Management. Not all vulnerabilities in software code put operators or users at real risk from software built using those vulnerable components, as the risk could be mitigated elsewhere or deemed to be negligible. One approach to managing this might be to communicate that software is "not affected" by a specific vulnerability through a Vulnerability Exploitability eXchange (or "VEX"), but other solutions may exist.

MITRE believes that the current work on this within the NTIA's consensus building work on Software Component Transparency is adequately addressing this item.

4. Flexibility of implementation and potential requirements. If there are legitimate reasons why the above elements might be difficult to adopt or use for certain technologies, industries, or communities, how might the goals and use cases described above be fulfilled through alternate means? What accommodations and alternate approaches can deliver benefits while allowing for flexibility?

MITRE has no comment to offer on this item.