Life Is On | Schneider Electric

Ms. Evelyn L. Remaley
Acting Administrator
National Telecommunications and Information Administration
U.S. Department of Commerce
1401 Constitution Avenue NW, Room 4725
Washington, DC 20230

June 17, 2021

RE: Request for Comments on Software Bill of Materials Elements and Considerations

Dear Acting Administrator Remaley:

Schneider Electric strongly supports the software transparency and security objectives behind the Request for Comments on Software Bill of Materials (SBOM) Elements and Considerations. As a critical manufacturer, we look forward to partnering with the Department on this important work.

At Schneider Electric, we drive digital transformation by integrating world-leading process and energy technologies, end-point to cloud connecting products, controls, software and services, across the entire lifecycle, enabling integrated company management, for homes, buildings, data centers, infrastructure and industries.

Our integrated solutions enable homes, commercial buildings, data centers, and critical infrastructure to operate more efficiently and securely. Our products and systems are used in over one million buildings worldwide, including 40,000 water & wastewater treatment installations, 40% of the world's hospitals, 10 of the world's top electric utilities, and 10 of the world's largest airports. The cybersecurity of these products and systems is therefore of vital importance to us and our customers.

Our responses to the RFC questions are listed below.

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

The elements described in the RFC are appropriate, but we recommend that the National Telecommunications and Information Administration (NTIA) consider the following additional elements as part of the "baseline component information":

a. **Date of SBOM creation and update**: We recommend adding a date field representing when the SBOM was created (in addition to the author name) as it is important to know the frequency of updates made and to explain possible evolutions linked to a component (e.g., license model changed, component versions changed, new dependencies created, etc.).

b. **Component licensing information**: We recommend explicitly stating when a component is open source versus proprietary. It is not obvious which components are open source or proprietary in a full SBOM. Providing this information would help consumers of SBOMs make informed decisions down to the component level.

c. **Component patch information**: We recommend providing information to verified users upon request regarding each release of the software, listing any components that have been patched in each release along with identifying the relevant patch applied. For embedded components with open vulnerabilities, the SBOM should also state whether these vulnerabilities are accessible by verified end users for patching.

d. **Unique reference id**: Consumption of SBOMs from multiple vendors, on the end customer side, can be automated only if there is a way to uniquely identify each component. A definition around this is lacking currently and hinders machine readability, automation, and efficiency. We provide a potential solution in our response to Question 3(a).

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

Many operational technology (OT) manufacturers and service providers follow the underlying principles of an SBOM through their conformance with existing international standards (e.g., IEC 62443 suite of standards for cybersecurity). Where possible, NTIA should map prospective SBOM requirements or "baseline component information" with corresponding requirements that appear in international standards. This will reinforce to the community that NTIA is not "creating" SBOM requirements, but simply leveraging best practices from the international standards community. This mapping could help increase adoption of SBOMs.

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**: A possible solution would be to define a large software globally unique identifier (GUID) space that would be assigned to organizations in large blocks, similar to MAC addresses, to use in conjunction with their current namespace approach. The assignment of GUIDs would be unique per component, and the version of the component would be appended to the end of the GUID as GUID#version. Depending on the needs, a vendor would register and request a block of namespaces with a central authority, similar to what the Institute of Electrical and Electronics Engineers (IEEE) does for MAC addresses, which can facilitate coordination among vendors.

b. **Software-as-a-Service and online services**:

i. Cloud-based software often has releases multiple times a day. Capturing an SBOM for each release (not to mention sharing it with customers) is simply not practical. A more practical approach would be to require that a new SBOM be generated whenever a component is updated or removed, or when a new component is introduced. SBOMs could be made

available to cloud consumers under a non-disclosure agreement (NDA) with cloud service providers.

ii. A complete BOM for a cloud system includes:
    a. An inventory of all "assets" used in the cloud system. This is called an "Asset Inventory" (ala ISO 27001 and other related standards). The Asset Inventory includes the Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) services used by the system and their security configuration (e.g., firewall rules, access control lists, etc.), as well as operating systems and other third-party software and frameworks hosted on the IaaS infrastructure. In a DevOps environment, assets change frequently, so the Asset Inventory is merely a "point in time" inventory of assets and thus has limited usefulness to a customer of the cloud system.
    b. An SBOM for applications developed by engineering teams and deployed to the cloud system. Like the Asset Inventory, in a DevOps environment the SBOM changes frequently so the SBOM is merely a "point in time" inventory of third-party software dependencies.

iii. A vendor's ISO 27001 or SOC2 compliance documentation can be used to determine if the vendor has the proper information security processes in place to continuously manage and mitigate risks, including risk due to software and infrastructure vulnerabilities.

c. **Legacy and binary-only software:** Legacy software, and any software that has had multiple releases, would need to create an SBOM for all released versions of the software since each SBOM is unique. The producer (supplier) would need to generate potentially hundreds of SBOMs for legacy software which is a significant undertaking for a company with hundreds or thousands of legacy products. In such cases, the code is essentially no longer maintainable. There are two problems here. One, for older, legacy embedded products and older software that are no longer maintained, the concern goes to the installed base; consumers should be encouraged to migrate to more modern devices, software, and firmware. Two, software using an older, no longer maintained component should: a) identify such components in the SBOM, and b) strive to replace such components within a reasonable period of time (not more than three years).

d. **Integrity and authenticity**: Signing of the SBOM can be accomplished but is currently not part of many build processes or related tools. Verifying the completeness of the SBOM is a challenge. Although the community can start small, we will still face the challenge of knowing when an SBOM is complete and understanding how to verify supplier claims around the SBOM.

e. **Threat model:** It should not be in the role or scope of the SBOM to support the detection of internal compromise. Other methods exist to address such a need, such as signing of the components used, and verification of that signature before including it in the build or loading it to run. However, the SBOM could include a field that indicates whether or not the component is signed (and verified) in order to aid with the Threat Model.

f. **High assurance use cases**: Once again, it is not the role of a Bill of Materials to address such points. Those belong in the component's technical specifications, which should be easily accessed on the component supplier's website using the component's GUID, as proposed earlier. Build environment specific data (Build BOM or Environment BOM) could be a supplementary to a

product BOM provided to the customer, if required, but should not be a part of the product bill of materials.

g. **Delivery**: Delivery will need to be through a publish/subscribe model to enable registration of SBOMs not included in the software/firmware package itself. Automation and service APIs would be necessary to sync consumers to the producer's SBOM database and the National Vulnerability Database. It will be critical for consumers to ensure they are referencing the proper software version SBOM as part of their asset management system. We suggest that SBOMs be made available to verified customers upon request.

h. **Depth**: It is not feasible for a software development company to include the full graph of the SBOM, as they will only have knowledge of the component they are using, which does not provide information on the sub-components used within. This information only exists within the component supplier's SBOM. Pulling all such SBOMs together to form a complete graph is not possible today. An SBOM will only supply the highest-level components used in the software.

i. **Vulnerabilities**: Once again, it is not the role of the SBOM to provide vulnerability data. In fact, since an SBOM is created at a specific point in time, it makes no sense to attempt to do so. The SBOM should simply provide the component identifier and version so that any associated vulnerabilities at a future point in time can be identified from a vulnerability database, such as the National Vulnerability Database.

j. **Risk Management**: This could be a viable approach for vulnerabilities in components listed in an SBOM. Associated risk could be zero (i.e., not affected), little, partial or full impact based upon the vulnerability.

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?***

SBOM transparency is important as both a producer and a consumer. There are, however, potential ramifications with the contents disclosed in an SBOM. For example, there are many design decisions that would keep a development team from using the "latest and greatest" software libraries or operating systems. To re-architect or upgrade components in products takes extensive development time and investment. Some customer agreements request that all components be "patched to the latest version", which may not be achievable due to restrictions in the hardware footprint, interoperability, licensing, or release schedules. A concern we have is that the SBOM would provide transparency without context for vendor decision-making around updates and information included or not included in the SBOM.

Schneider Electric sincerely appreciates the opportunity to comment on the RFC. If you have any questions or need additional information, please contact me at Patrick.Ford@se.com.

Sincerely,

Patrick M. Ford

Patrick M. Ford
Regional Chief Information Security Officer, Americas Region
Schneider Electric