kaspersky

Comments Regarding the Software Bill of Materials Elements and Considerations

Kaspersky's Submission

June 2021

Kaspersky is grateful for the opportunity to provide comments on the Software Bill of Materials (SBOM) elements and further considerations.

Before addressing the questions outlined in the document (www.govinfo.gov/content/pkg/FR-2021-06-02/pdf/2021-11592.pdf), we would like to share the methodology applied at Kaspersky for compiling SBOM and for tracking updates/new versions for components of third parties. We apply an automated approach, also for our vulnerability management, and this includes:

1. Each product component has meta-information associated with it, and it also has information about third-party software the component uses. Here's an example:
   <externals>
   <usage name="tinyxml" version="2.6.2" path="" type="source" comment="" />
   <usage name="zlib" version="1.2.11" type="source" comment="(C) 1995-2017 Jean-loup Gailly and Mark Adler (http://zlib.net/)" />
   <usage name="wixtoolset" version="3.11.1.2318" type="source" comment="Copyright (c) .NET Foundation and contributors (http://wixtoolset.org/about/license/)" />
   </externals>

   This information is updated manually when new or updated third-party software is integrated into the component. We document and provide the following baseline elements: supplier name, component name, unique identifier, version string, component hash. and relationship.

2. A product list of third-party dependencies is composed by combining the lists for every component the product uses.
3. At Kaspersky we also have an internal automated service, which uses the data from components' meta-files to identify if any dependency has a newer version, and therefore should be updated. If a newer version is indeed available, the service creates a ticket for a development team to integrate the newer version. The third-parties are updated both for a newly developed version and for already released products.
4. SBOM is also used to generate legal notices file that shows license information for third-party software.

kaspersky

Below, we take the opportunity to share our thoughts and suggestions to the questions indicated in the public consultation:

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

"Baseline component information" is sufficient to cover most use cases we consider at Kaspersky. Sometimes we encounter difficulties with software identification based on the fields (Supplier name, Component name, Version of the component), but this can easily overcome by requesting additional information from the vendor. A component hash is useful only for components that are distributed and integrated in binary format. If a component is distributed in source code, each build of the component by any vendor would have a different hash.

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

At Kaspersky, SBOM use cases are divided into two groups, which could be also applied for the document: (a) for vendors, and (b) for consumers.

The use cases for us as a vendor include:

- third-party component identification and tracking;
- SBOM generation;
- SBOM distribution;
- SBOM updates; and
- Vulnerability exchange (VEX) process.

The use cases for us as a consumer includes:

- SBOM discovery and processing;
- Checking vulnerability sources for SBOM elements; and
- Tracking and resolving vulnerabilities for SBOM elements (including consuming VEX).

3. Issues that should be considered in defining SBOM elements today and in the future.

a. Software Identity.

Difficulties with software identification create some problems, but this should not be considered a key challenge and should not stop SBOM implementation. There could be rare cases when there are issues with software identification being erased, so we provide URLs from the component that was downloaded. These URLs serve as an extra identifier for the component and can be later used for checking the updates.

b. Software-as-a-service and online services.

Companies use software-as-a-service to shift the cybersecurity burden to the service provider (in most cases SaaS does not require any software installation on the customer network). The same idea is proposed in Exec. Order No. 14,028, Section 3. if a company uses software-as-a-service it reduces cyber risk exposure. In case of SaaS, the customer might delegate SBOM processing to the SaaS vendor.

e. Threat model:

SBOM is usually generated automatically during the software build and on the same infrastructure. This means that it has the same trusted level as the software itself and, therefore, cannot be used to check the integrity of the systems used to build the software component.

f. High assurance use cases:

Practices described in Executive Order No. 14028 § 4(e)(i)–(x) are related to vendor uses cases. The outlined practices will form some criteria to verify the trustworthiness of the "critical software". The practices include SBOM generation as well as other practices to produce secure and trusted software. Saying that, we believe that SBOM elements are sufficient, and the practices specified § 4(e)(i)–(x) outline a broader process where SBOM generation is one of the outputs.

To discuss the contents of this submission and/or request further information, please contact Igor Kumagin, cybersecurity expert at Kaspersky (igor.kumagin@kaspersky.com, +7 965 193 5989).