Software Bill of Materials

Transparency in the Software Supply Chain

Information Session for the Energy Community
January 26, 2021
Overview

- Background
  - The case for transparency
  - Why this is important
- What is an SBOM
  - Some of the work has already been done
  - A focus on the basics
- Why should we SBOM
  - Understanding the value
  - Use cases for the energy sector
- Some lessons from the other sectors’ “proof of concept” exercises
- Next steps for the energy community

**Takeaways:**
* SBOM will be an invaluable tool for managing cybersecurity and software supply chain risk.
* The energy and power community can safely experiment with this tool through a "proof of concept" exercise.
Motivating Example #1

Amnesia: 33 vulnerabilities impact millions of smart and industrial devices

Security researchers have identified 33 security flaws in four open-source TCP/IP stacks used across a wide range of smart products.
Motivating Example #1

Amnesia: 33 vulnerabilities impact millions of smart and industrial devices

Security researchers have identified 33 security flaws in four open-source TCP/IP stacks used across a wide range of smart products.

Can any organization that makes or uses software easily answer:

Am I affected by $vulnerability$ ?
Motivating Example #2

Researchers at security firm JSOF were forced to scour LinkedIn to identify companies that might use the vulnerable Treck IP library to disclose Ripple20 risks.
Transparency can help markets thrive

• Food ingredients and food labels
• Safety Data Sheets in the chemical industry
• Hardware Bills of Material (BOM) in industry
• Naming and tracking components can drive innovation (e.g. CVE)
Software Supply Chain
Software Supply Chain
What is an SBOM?

A dependency tree
What is an SBOM?

A Software Bill of Materials (SBOM) is effectively a list of ingredients or a nested inventory.

It is “a formal record containing the details and supply chain relationships of various components used in building software"
What is an SBOM?

Supplier Component Version Hash

Included in

Known Unknowns

Bob’s Browser v2.2
unknown

Included in

Known

Acme Application v1.1

Included in

known

Carol’s Compression Engine v3.1

Included in

root

Bingo Buffer v2.1

unknown
Why aren’t we doing this today?

• Licensing concerns and open source restrictions
• It’s a chicken-and-egg problem.
• It’s hard: benefits require machine readability for automation.
• It’s complex: involves integrating some technical and operational innovation.
• Success requires non-trivial adoption.
SBOMs support multiple use cases across the software and security world.
Open source
Middleware
Commercial SW
Embedded
Customers

NTIA’s Process on Software Component Transparency

Cross sector

Open to all Stakeholders
Multistakeholder Characteristics
Bottom up process
Accountable
Transparent
Consensus Driven

NTIA’s open, transparent, consensus-based processes bring together diverse stakeholders, and can catalyze real progress across the ecosystem.
What we’re not doing

• Building out regulation
• Source code disclosure
• Standards development
• Solving all supply chain or assurance issues
Beyond talking

- From descriptive work to implementation
- Scalability, automation, and interoperability
The Future of SBOM

• Interest in the US and around the world
• SBOM in standards and guidance
• It is critical to establish a common set of practices and market expectations that is viable and reflects the needs of industry.
Summing up

• Software Bill of Materials is a technical and operational model of tracking software dependencies.

• SBOMs enable better software security and supply chain risk management
  • Vulnerability Management
  • Procurement
  • Dealing with emerging risks

• While we need cross-sector solutions, each community will need to understand its own unique implementation.

• Need continued industry leadership to guide investment, standards, and policy around the world.

• More information
  • Published documents: ntia.gov/SBOM
  • About the SBOM process: ntia.gov/SoftwareTransparency
  • Reach out to get involved: afriedman@ntia.gov
Next steps for the Energy Community

• More detailed briefings in February / March
  • Highlight the global consensus on SBOM structure and implementation
  • Existing technical standards
  • How to experiment with SBOM: lessons from healthcare

• Initial conversations about potential structures of proof-of-concept exercise

• Search for initial participants

• Want to stay involved? Have more questions?
  afriedman@ntia.gov
Three formats to implement SBOM

**SPDX** is an open standard for communicating software bill of material information (including components, licenses, copyrights, and security references). The SPDX specification is developed by the SPDX workgroup, which is hosted by The Linux Foundation. The grass-roots effort includes representatives from more than 20 organizations—software, systems and tool vendors, foundations and systems integrators.

**CycloneDX** is a software bill of materials (SBOM) standard, purpose-built for software security contexts and supply chain component analysis. The specification is maintained by the CycloneDX Core working group, with origins in the OWASP community.

**SWID** tags record unique information about an installed software application, including its name, edition, version, whether it is part of a bundle and more. SWID tags support software inventory and asset management initiatives. The structure of SWID tags is specified in international standard ISO/IEC 19770-2:2015.

- We have identified the common elements.
- A ‘multilingual’ ecosystem does not offer too many challenges
- Rather than pick a winner, we will build out guidance to support all formats with effective interoperability.
# Implementing core SBOM fields

<table>
<thead>
<tr>
<th>Field</th>
<th>SPDX</th>
<th>SWID</th>
<th>CycloneDX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supplier</td>
<td>(3.5) PackageSupplier:</td>
<td>&lt;Entity&gt; @role (softwareCreator/publisher), @name</td>
<td>publisher</td>
</tr>
<tr>
<td>Component</td>
<td>(3.1) PackageName:</td>
<td>&lt;softwareIdentity&gt; @name</td>
<td>name</td>
</tr>
<tr>
<td>Unique Identifier</td>
<td>(3.2) SPDXID:</td>
<td>&lt;softwareIdentity&gt; @tagID</td>
<td>bom/serialNumber and component/bom-ref</td>
</tr>
<tr>
<td>Version</td>
<td>(3.3) PackageVersion:</td>
<td>&lt;softwareIdentity&gt; @version</td>
<td>version</td>
</tr>
<tr>
<td>Component Hash</td>
<td>(3.10) PackageChecksum:</td>
<td>&lt;Payload&gt;/../&lt;File&gt; @[hash-algorithm]:hash</td>
<td>hash</td>
</tr>
<tr>
<td>Relationship</td>
<td>(7.1) Relationship: CONTAINS</td>
<td>&lt;Link&gt;@rel, @href</td>
<td>(Nested assembly/subassembly and/or dependency graphs)</td>
</tr>
<tr>
<td>SBOM Author</td>
<td>(2.8) Creator:</td>
<td>&lt;Entity&gt; @role (tagCreator), @name</td>
<td>bom-descriptor: metadata/manufacture/contact</td>
</tr>
</tbody>
</table>