NTIA SBOM: Formats and Tooling

JC Herz and Kate Stewart
Technical Briefing for the Energy Community
Formats and Tooling: Objectives  (1/3)

● Focus on enabling automated SBOM generation and use
● Build and expand on what already exists
● Try to avoid re-inventing the wheel
● Document what tools exist
● Identify any gaps in the tooling ecosystem
● Promote inclusion of SBOM capabilities into existing tools and services
Formats and Tooling: Objectives (2/3)

Identify SBOM Formats in Commercial Use
- SPDX - https://spdx.github.io/spdx-spec/
- CycloneDX - https://cyclonedx.org/docs/1.2/

Identify Software Identifiers in Commercial Use and Emerging Identifiers
- Common Package Enumeration - CPE
- Package URLs - PURL
- Software ID tags - SWID tag
- Software Heritage persistent ID - SWHID
Formats and Tooling: Objectives  (3/3)

- Define and categorize criteria for the minimum required information in an SBOM
  - Field definitions
  - Data extensions for provision of additional/external/deeper information

- Enable translation between SBOM formats
  - “Decoder Ring” tool - in progress
  - “SwiftBOM” tool - in progress, used in HealthCare PoC

- Create Playbooks for Generation and Consumption of SBOM
  - Supplier Playbook - in progress
  - Consumer Playbook - draft release:
    https://docs.google.com/document/d/1Ae0l1MDS8m1on58e8mdVIA9NujzPD0k5j352VIDZr9I/edit
What should a minimum viable SBOM contain?

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

Current SBOM Options Available

### SPDX

*File formats:* .xls, .spdx, .rdf, .json, .yml, .xml

### SWID

*File formats:* .xml

### CycloneDX

*File formats:* .json, .xml
Translating between SBOM formats & file formats

SwiftBOM:  (SPDX(.spdx), SWID(.xml), CycloneDX(.xml,.json))
- Demo at:  https://democert.org/sbom/
- Source code at:  https://github.com/CERTCC/SBOM/tree/master/sbom-demo

DecoderRing:  (SPDX (.spdx), SWID(.xml))
- Source code at:  https://github.com/DanBeard/DecoderRing

SPDX tools:  ( SPDX (.spdx, json, yaml, rdf, xml, xls) )
- Demo at:  https://tools.spdx.org/app/
- Source code at:  https://github.com/spdx/spdx-online-tools
Where use an SBOM? All stages

Upstream / Produce

Open Source Projects  3rd Party

Need for reference tooling for efficient and effective exchange of software bills of materials to enable compliance, security, export control, pedigree and provenance workflows.

Development → Build → Test → Packaging → Policy Checks → Release/Operate

Downstream / Consume

Customers

Open Source Projects  Products  Services
### Taxonomy used for Classifying SBOM Tools
(UPDATED Feb26)

<table>
<thead>
<tr>
<th>Category</th>
<th>Type of Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Produce</td>
<td>Build</td>
<td>SBOM is automatically created as part of building a software artifact and contains information about the build.</td>
</tr>
<tr>
<td></td>
<td>Manual</td>
<td>A person will manually fill in the SBOM information</td>
</tr>
<tr>
<td></td>
<td>Analysis</td>
<td>Analysis of source or binary files will generate the SBOM by inspection of the artifacts and any associated sources.</td>
</tr>
<tr>
<td>Consume</td>
<td>View</td>
<td>Be able to understand the contents in human readable form (picture, figures, tables, text.). Use to support decision making &amp; business processes.</td>
</tr>
<tr>
<td></td>
<td>Diff</td>
<td>Be able to compare two documents of a given formation and clearly see the differences. For instance, comparing between two versions of a piece of software.</td>
</tr>
<tr>
<td></td>
<td>Import</td>
<td>Be able to import an SBOM into your system for further processing</td>
</tr>
<tr>
<td>Transform</td>
<td>Translate</td>
<td>Change from one file type to another file type while preserving the same information.</td>
</tr>
<tr>
<td></td>
<td>Merge</td>
<td>Multiple sources of documents can be merged together for analysis and audit purposes</td>
</tr>
<tr>
<td></td>
<td>Tool support</td>
<td>Support use in other tools by APIs, object models, libraries, or other reference sources.</td>
</tr>
</tbody>
</table>
## Taxonomy used for Classifying SBOM Tools

<table>
<thead>
<tr>
<th>Category</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Produce</td>
<td>Build</td>
<td>Document is automatically created as part of building an artifact and contains information about the build.</td>
</tr>
<tr>
<td></td>
<td>Manual</td>
<td>A person will manually fill in the information</td>
</tr>
<tr>
<td></td>
<td>Audit Tool</td>
<td>A source code analysis or audit tool will generate the document by inspection of the artifact and any associated sources.</td>
</tr>
<tr>
<td>Consume</td>
<td>View</td>
<td>Be able to understand the contents in human readable form (picture, figures, tables, text.). Use to support decision making &amp; business processes.</td>
</tr>
<tr>
<td></td>
<td>Diff</td>
<td>Be able to compare two documents of a given formation and clearly see the differences. For instance, comparing between two versions of a piece of software.</td>
</tr>
<tr>
<td></td>
<td>Analyze</td>
<td>Be able to import a document into your system</td>
</tr>
<tr>
<td>Transform</td>
<td>Translate</td>
<td>Change from one file type to another file type while preserving the same information.</td>
</tr>
<tr>
<td></td>
<td>Merge</td>
<td>Multiple sources of documents can be merged together for analysis and audit purposes</td>
</tr>
<tr>
<td></td>
<td>Tool integration</td>
<td>Support use in other tools by APIs, libraries.</td>
</tr>
</tbody>
</table>
Tool Support for Different SBOM Formats

**SPDX**

1. Format Overview
2. Format Publishing History
3. Tool Classification Taxonomy

**Open Source Tools**
- Atrib
- ICSE:
- Java
- Fortify
- Open Source Software Review Toolkit (OSSRT)

**Proprietary Products**
- CyberTrack
- RS232
- MedScan

**Proprietary Products**
- Infoprice
- Source Auditor
- TrueSource
- Tightrope

---

**SWID**

1. Format Overview
2. Format Publishing History
3. Tool Classification Taxonomy

**Open Source Tools**
- SWIDgen
- SWID Tool Generator
- SWID Maven Plugin

**Proprietary Products**
- IT Operations Management
- CyberTrack
- AWS

**Proprietary Products**
- Source Auditor
- TrueSource
- Tightrope

---

**CycloneDX**

1. Format Overview
2. Format Publishing History
3. Tool Classification Taxonomy

**Open Source Tools**
- CycloneDX Core
- CycloneDX for .NET
- CycloneDX for Java

**Proprietary Products**
- SAST
- DAST
- DAST Dependency-Track

**Proprietary Products**
- Source Auditor
- TrueSource
- Tightrope

---

http://tiny.cc/SPDX
http://tiny.cc/SWID
http://tiny.cc/CycloneDX
SBOMs Examples (Work in Progress)

SPDX
- https://github.com/swinslow/spdx-examples - source & binary examples

CycloneDX
- https://github.com/CycloneDX/sbom-examples - binary examples

SWID
- Time 1.9 from Red Hat distro - binary example
SBOM Playbooks: Supplier Playbook

- Supplier defined to include: commercial vendor, contract developer, open source software supplier developing and maintaining OSS code.
- SBOM production workflow: development pipeline vs. legacy processes
- SBOM scope: What’s in the Box
- Build Artifacts
  - Functional workflow (tool-agnostic) for commit → build with SBOM production as an output
  - Example outputs: SPDX, CycloneDX
- Provision of SBOMs to recipients
  - Reference to NTIA Framing Group report:
SBOM Playbook: Consumer Playbook

- Acquisition of SBOM from supplier
- SBOM Ingestion and Parsing
- Software Entity Resolution
- Data Flows into Third Party Processes and Platforms
  - Configuration Management Database
  - Security Operations Center
  - Software Asset Management System
- Ongoing Monitoring
Areas to Learn: Generalized vs. Energy-Specific Requirements

- Generalized requirements for code: software, firmware, embedded
- Where do SBOM requirements of firmware/embedded diverge from IT?
- Where do SBOM requirements for licensed/proprietary third party components diverge from third party open source components?
- Lessons Learned and Best Practices for SBOM access control
  - Open Formats
  - Content may be delivered under NDA
  - Rely on confidentiality terms and access control, rather than IP/copyright
- Why this matters: SBOM is an intermediary phase of the data
  - Operational requirement for data to be ingested by enterprise processes and platforms
  - Ex: CMDB, SAM, SOC
  - Configuration management can’t become a “derivative work” and function as intended.