Formats & Tooling Working Group

Co-chairs: JC Herz & Kate Stewart
Meeting biweekly since July 2018

- Fridays at 1100 EDT
- [https://lists.linuxfoundation.org/mailman/listinfo/ntia-sbom-formats](https://lists.linuxfoundation.org/mailman/listinfo/ntia-sbom-formats)

Adoption of concepts from framing into existing formats and tools that work with those formats.
Agenda

• Workgroup Goals
• Recap of Formats in Use
  • Populating Example repos, Ecosystem Documents
• Playbooks
  • Consumer Playbook Overview
  • Supplier Playbook Overview
• Future Directions
• Feedback Requests
Formats and Tooling Workgroup Goal

Wrapping up from phase I, we identified for the need for:

- **Tooling**
  - Documenting tooling
  - Identifying tooling gaps
  - Documenting processes ← Playbooks starting to address
  - Turnkey universal translation tools

Formats and Tooling workgroup is focusing on addressing these items.
Formats and Tooling: Objectives

Identify SBOM Formats in Commercial Use

- SPDX - [https://spdx.github.io/spdx-spec/](https://spdx.github.io/spdx-spec/)
- CycloneDX - [https://cyclonedx.org/docs/1.2/](https://cyclonedx.org/docs/1.2/)

Identify Software Identifiers in Commercial Use and Emerging Identifiers

- Common Platform Enumeration - [CPE](https://cpe.mitre.org/)
- Package URLs - [PURL](http://purl.org/net/purl)
- Software ID tags - [SWID tag](https://www.swid.org)
- Software Heritage persistent ID - [SWHID](https://swhdb.org)

Formats and Tooling: Objectives

- Define and categorize criteria for the minimum required information in an SBOM from Framing Definitions
  - 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 - draft release: [https://docs.google.com/document/d/16FwpPX3P0Pd1D82Dp2VmpRnaMWUA-wvfXbL7AI XDthM/edit](https://docs.google.com/document/d/16FwpPX3P0Pd1D82Dp2VmpRnaMWUA-wvfXbL7AI XDthM/edit)
  - Consumer Playbook - draft release: [https://docs.google.com/document/d/1Ae0l1MDS8m1on58e8mdVIA9NujzPD0k5j352VlDZr9I/edit](https://docs.google.com/document/d/1Ae0l1MDS8m1on58e8mdVIA9NujzPD0k5j352VlDZr9I/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><strong>Supplier Name</strong></td>
<td>(3.5) PackageSupplier:</td>
<td>&lt;Entity&gt; @role (softwareCreator/publisher), @name</td>
<td>publisher</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Component Name</strong></td>
<td>(3.1) PackageName:</td>
<td>&lt;softwareIdentity&gt; @name name</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Unique Identifier</strong></td>
<td>(3.2) SPDXID:</td>
<td>&lt;softwareIdentity&gt; @tagID bom/serialNumber and component/bom-ref</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Version String</strong></td>
<td>(3.3) PackageVersion:</td>
<td>&lt;softwareIdentity&gt; @version version</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Component Hash</strong></td>
<td>(3.10) PackageChecksum:</td>
<td>&lt;Payload&gt;/../&lt;File&gt; @[hash-algorithm]:hash hash</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Relationship</strong></td>
<td>(7.1) Relationship: CONTAINS</td>
<td>&lt;Link&gt; @rel, @href (Nested assembly/subassembly and/or dependency graphs)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Author Name</strong></td>
<td>(2.8) Creator:</td>
<td>&lt;Entity&gt; @role (tagCreator), @name bom-descriptor:metadata/manufacture/contact</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

CycloneDX CLI: ( CycloneDX (.xml, .json), SPDX(.spdx))
- Source code at: https://github.com/CycloneDX/cyclonedx-cli
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

- Open Source Projects
- Products
- Services

Customers
<table>
<thead>
<tr>
<th>Category</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Produc</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>Consum</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>Transfor</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>
## Information to Collect per Tool

### Tool Template

<table>
<thead>
<tr>
<th>Support</th>
<th>Produce?, Consume?, Transform?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td></td>
</tr>
<tr>
<td>Location</td>
<td>Website: Source:</td>
</tr>
<tr>
<td>Installation instructions</td>
<td></td>
</tr>
<tr>
<td>How to use</td>
<td></td>
</tr>
<tr>
<td>Versions Supported</td>
<td></td>
</tr>
</tbody>
</table>

### Example: FOSSology

<table>
<thead>
<tr>
<th>Support</th>
<th>Produce (Audit tool, Manual), Consume(View, Diff, Analyze), Transform(Translate, Merge, Tool Integration)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>FOSSology is an open source license compliance software system and toolkit allowing users to run license, copyright and export control scans from a REST API. As a system, a database and web UI are provided to provide a compliance workflow. As part of the toolkit multiple license scanners, copyright and export scanners are tools available to help with compliance activities.</td>
</tr>
</tbody>
</table>
| Location | Website: [https://www.fossology.org/](https://www.fossology.org/)  
Source: [https://github.com/fossology](https://github.com/fossology) |
| Installation instructions | [https://www.fossology.org/get-started/](https://www.fossology.org/get-started/) |
| How to use | [https://www.fossology.org/get-started/basic-workflow/](https://www.fossology.org/get-started/basic-workflow/) |
| Versions Supported | SPDX 2.1, SPDX 2.2 |
## Tool Support for Different SBOM Formats

<table>
<thead>
<tr>
<th>SPDX</th>
<th>SWID</th>
<th>CycloneDX</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Format Overview</strong>&lt;br&gt;2</td>
<td><strong>Format Overview</strong>&lt;br&gt;2</td>
<td><strong>Format Overview</strong>&lt;br&gt;2</td>
</tr>
<tr>
<td>SPDX Publishing History</td>
<td>SWID Publishing History</td>
<td>CycloneDX Publishing History</td>
</tr>
<tr>
<td>SPDX Tool Classification Taxonomy</td>
<td>SWID Tool Classification Taxonomy</td>
<td>CycloneDX Tool Classification Taxonomy</td>
</tr>
</tbody>
</table>

### Open Source Tools

<table>
<thead>
<tr>
<th>SPDX</th>
<th>SWID</th>
<th>CycloneDX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aqua</td>
<td>SWID</td>
<td>CycloneDX Core W-Java</td>
</tr>
<tr>
<td>HCLTool</td>
<td>SWID</td>
<td>CycloneDX for .NET</td>
</tr>
<tr>
<td>Jet</td>
<td>SWID</td>
<td>CycloneDX for NPM</td>
</tr>
<tr>
<td>Maven</td>
<td>SWID</td>
<td>CycloneDX for Maven</td>
</tr>
<tr>
<td>Nexus</td>
<td>SWID</td>
<td>CycloneDX for Gradle</td>
</tr>
<tr>
<td>Nexus Repository Manager</td>
<td>SWID</td>
<td>CycloneDX for PHP+Composer</td>
</tr>
<tr>
<td>Open Source Software Review Toolkit (OSRT)</td>
<td>SWID</td>
<td>CycloneDX for Python</td>
</tr>
<tr>
<td>OWASP Dependency-Track</td>
<td>SWID</td>
<td>CycloneDX for Ruby Gems</td>
</tr>
<tr>
<td>OWASP Dependency-Tracker</td>
<td>SWID</td>
<td>CycloneDX for Rust Cargo</td>
</tr>
<tr>
<td>OWASP Dependency-Track</td>
<td>SWID</td>
<td>CycloneDX for SBT</td>
</tr>
<tr>
<td>OWASP Dependency-Track</td>
<td>SWID</td>
<td>CycloneDX for ELT Mix</td>
</tr>
<tr>
<td>OWASP Dependency-Tracker</td>
<td>SWID</td>
<td>CycloneDX for Elastic Search</td>
</tr>
<tr>
<td>OWASP Dependency-Tracker</td>
<td>SWID</td>
<td>CycloneDX for Go</td>
</tr>
</tbody>
</table>

### Proprietary Products

<table>
<thead>
<tr>
<th>SPDX</th>
<th>SWID</th>
<th>CycloneDX</th>
</tr>
</thead>
<tbody>
<tr>
<td>CyberTrack</td>
<td>SWID</td>
<td>OWASP Dependency-Tracker Docker Image</td>
</tr>
<tr>
<td>COAB</td>
<td>SWID</td>
<td>OWASP Dependency-Tracker Docker Image</td>
</tr>
<tr>
<td>CyberSentry</td>
<td>SWID</td>
<td>OWASP Dependency-Tracker Docker Image</td>
</tr>
<tr>
<td>Direct2D</td>
<td>SWID</td>
<td>OWASP Dependency-Tracker Docker Image</td>
</tr>
<tr>
<td>NextStep</td>
<td>SWID</td>
<td>OWASP Dependency-Tracker Docker Image</td>
</tr>
<tr>
<td>Proteus</td>
<td>SWID</td>
<td>OWASP Dependency-Tracker Docker Image</td>
</tr>
<tr>
<td>Perimeter</td>
<td>SWID</td>
<td>OWASP Dependency-Tracker Docker Image</td>
</tr>
<tr>
<td>SourceAuditor</td>
<td>SWID</td>
<td>OWASP Dependency-Tracker Docker Image</td>
</tr>
<tr>
<td>TrueSource</td>
<td>SWID</td>
<td>OWASP Dependency-Tracker Docker Image</td>
</tr>
<tr>
<td>Vantage</td>
<td>SWID</td>
<td>OWASP Dependency-Tracker Docker Image</td>
</tr>
</tbody>
</table>

### Other Resources

- [SPDX](http://tiny.cc/SPDX)
- [SWID](http://tiny.cc/SWID)
- [CycloneDX](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 Reference Corpus (Work in Progress)

Proposing set of Projects to generate source and binary SBOMs for in the different formats for the same example, to aid compare & contrast.

- Basic: Hello World, Blinky, Time
- Intermediate: Juice-Shop, WebGoat, NodeGoat, vscode
- Advanced: <container tbd>, openAPS

To participate in selection or make suggestions, add comments in working document: NTIA SBOM Reference Corpus
Playbooks for using “Tools in Operation”

- Concepts of Operation (CONOPS) for how they can be used
  - Generation and Consumption
  - Different Use Cases
    - Software Lifecycle Management
    - Entitlements
    - Vulnerability Management
  - Different Roles in the Supply Chain
    - Third Party Supplier (OSS, Commercial Software)
    - Integrator
    - First-party Developer (Internal Enterprise DevOps)
    - Procurement
    - Compliance (interface with external certifiers, regulators, insurers)
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
• IP and Confidentiality Status of SBOMs and Underlying Data
  • Everyone except the brand owner is an intermediary supplier - the wrong set of rules for data provision thwarts transparency and security (the broken Christmas lightbulb problem)
• Question for Auto-ISAC: Files vs. Flows/Channels (SBOM/DBOM)
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
  • Areas of consensus: single application and its compiled dependencies
  • Still in discussion: external services (SBOM formats can do this)
  • Need for clarity about SBOM coverage: runtime dependencies, container contents
  • As long as extent of coverage is clear (i.e. fields present with “no attestation”), level of detail will ultimately be negotiated between supplier and consumer
• 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:
• IP Status of SBOMs: not making SBOMs carry the weight of contract enforcement, and use of confidentiality vs. copyright
Areas to Learn: Generalized vs. Industry-Specific Requirements

- Generalized requirements for code: software, firmware, embedded
- Where do SBOM requirements of firmware/embedded diverge from IT?
  - Ex: Auto industry, Energy, Medical devices with firmware and embedded
- Where do SBOM requirements for licensed/proprietary third party components diverge from third party open source components?
- Lessons Learned and Best Practices for SBOM IP
  - Open Formats
  - Content may be delivered under NDA
  - Content must be capable of transfer to final-goods-assembler without copyright restriction
    - Assumption: NDAs carry the weight of confidentiality terms
- 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.
Next Steps

- Continue to collect tools (Know a tool to be added to each ecosystem document?) Put a comment in the document, so it can be added.
  - **SWID:** [http://tiny.cc/SWID](http://tiny.cc/SWID)
  - **SPDX:** [http://tiny.cc/SPDX](http://tiny.cc/SPDX)
  - **CycloneDX:** [http://tiny.cc/CycloneDX](http://tiny.cc/CycloneDX)

- Continue population of examples
  - Associated with each format
  - Reference corpus of examples illustrated with each format

- Finalize Playbooks
  - **Consumer Playbook Draft:**
    [https://docs.google.com/document/d/1Ae0l1MDS8m1on58e8mdVIA9NuizPD0k5i352VIDZr9I/edit](https://docs.google.com/document/d/1Ae0l1MDS8m1on58e8mdVIA9NuizPD0k5i352VIDZr9I/edit)
  - **Supplier Playbook Draft:**
    [https://docs.google.com/document/d/1Ae0l1MDS8m1on58e8mdVIA9NuizPD0k5i352VIDZr9I/edit](https://docs.google.com/document/d/1Ae0l1MDS8m1on58e8mdVIA9NuizPD0k5i352VIDZr9I/edit)

- Collaboration with medical and any new PoCs, provide feedback of gaps to framing

Volunteers interested on working on above areas? Feedback on proposed approach?
More Info...

**Meetings:** Every 2 weeks, next meeting scheduled for **Jan 22 at 11am EST.** Contact leads to be added to meeting invite

**Mailing List:** [ntia-sbom-formats@linuxfoundation.org](mailto:ntia-sbom-formats@linuxfoundation.org)

**Subscribe at:** [https://lists.linuxfoundation.org/mailman/listinfo/ntia-sbom-formats](https://lists.linuxfoundation.org/mailman/listinfo/ntia-sbom-formats)

**Shared Drive:**
[https://drive.google.com/drive/folders/1KAQ7AWlWMKcSFnRc_S-7XB76xFRRWLMt](https://drive.google.com/drive/folders/1KAQ7AWlWMKcSFnRc_S-7XB76xFRRWLMt)