CISA SBOM-A-RAMA

DEC 15TH PRESENTATIONS
JOSH CORMAN
@JOSHCORMAN
@IAMTHECAVALRY

HISTORY OF SBOM

(SEE VIDEO)
Hello, world!
Produce

The person or organization that creates a software component or software for use by others

[write/create/assemble/package]
Produce

The person or organization that creates a software component or software for use by others

[write/create/assemble/package]

Choose

The person or organization that decides the software, products, and/or suppliers for use

[purchase/acquire/source/select/approve]
Produce
The person or organization that creates a software component or software for use by others
[write/create/assemble/package]

Choose
The person or organization that decides the software, products, and/or suppliers for use
[purchase/acquire/source/select/approve]

Operate
The person or organization that operates the software component or software
[uses.monitor/maintain/defend/respond]
### Benefits

<table>
<thead>
<tr>
<th>Produce</th>
<th>Choose</th>
<th>Operate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Less unplanned, unscheduled work</td>
<td>A more accurate total cost of ownership</td>
<td>More efficient administration</td>
</tr>
<tr>
<td>Avoid known vulnerabilities</td>
<td>Easier due diligence</td>
<td>Faster identification and resolution. Know if and where specific software is affected.</td>
</tr>
<tr>
<td>Quantify and manage licenses and associated risk</td>
<td>Easier due diligence</td>
<td>More efficient, accurate response to license claims</td>
</tr>
<tr>
<td>Easier risk evaluation. Identify compliance requirements earlier in lifecycle</td>
<td>More accurate due diligence, catch issues earlier in lifecycle</td>
<td>Streamlined process</td>
</tr>
<tr>
<td>Make assertions about artifacts, sources, and processes used</td>
<td>Make informed, attack-resistant choices about components</td>
<td>Validate claims under changing and adversarial conditions</td>
</tr>
<tr>
<td>Benefits</td>
<td>Produce</td>
<td>Choose</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td><strong>Cost</strong></td>
<td>Less unplanned, unscheduled work</td>
<td>A more accurate total cost of ownership</td>
</tr>
<tr>
<td><strong>Security Risk</strong></td>
<td>Avoid known vulnerabilities</td>
<td>Easier due diligence</td>
</tr>
<tr>
<td><strong>License Risk</strong></td>
<td>Quantify and manage licenses and associated risk</td>
<td>Easier due diligence</td>
</tr>
<tr>
<td><strong>Compliance Risk</strong></td>
<td>Easier risk evaluation. Identify compliance requirements earlier in lifecycle</td>
<td>More accurate due diligence, catch issues earlier in lifecycle</td>
</tr>
<tr>
<td><strong>High Assurance</strong></td>
<td>Make assertions about artifacts, sources, and processes used</td>
<td>Make informed, attack-resistant choices about components</td>
</tr>
</tbody>
</table>
Ecosystem, Network Effects, and Public Health Benefits
Depth vs. Effectiveness

Limited visibility enables less awareness of risk

More complete visibility enables more complete awareness of risk
Entire Supply Chain
Parts

Compound Parts

Final Goods Assembled

Operators & Consumers

Cross Sector
Thank you!
References

- NTIA Software Bill of Materials Website
  https://www.ntia.gov/sbom

- Roles and Benefits for SBOM Across the Supply Chain
CASSIE CROSSLEY
PRODUCT & SYSTEMS SECURITY DIRECTOR AT SCHNEIDER ELECTRIC

PRACTITIONER PERSPECTIVE
(SEE VIDEO)
JENNINGS ASKE
CISO, NEW YORK-PRESBYTERIAN HOSPITAL

PRACTITIONER PERSPECTIVE
(SEE VIDEO)
Framing Software Component Transparency: Establishing a Common Software Bill of Materials (SBOM)

https://tinyurl.com/5n9b45sv

Art Manion <amanion@cert.org>
What is an SBOM?

An SBOM is a formal, machine-readable inventory of software components and dependencies, information about those components, and their hierarchical relationships. These inventories should be comprehensive – or should explicitly state where they could not be. SBOMs may include open source or proprietary software and can be widely available or access-restricted.

Global model: SBOM elements, baseline attributes, processes, terminology
<table>
<thead>
<tr>
<th>Attribute</th>
<th>SPDX</th>
<th>CycloneDX</th>
<th>SWID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author Name</td>
<td>(2.8) Creator:</td>
<td>metadata/authors</td>
<td>&lt;Entity&gt; @role</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(tagCreator), @name</td>
</tr>
<tr>
<td>Timestamp</td>
<td>(2.9) Created:</td>
<td>metadata/timestamp</td>
<td>&lt;Meta&gt;</td>
</tr>
<tr>
<td>Supplier Name</td>
<td>(3.5) PackageSupplier:</td>
<td>metadata/supplier</td>
<td>&lt;Entity&gt; @role</td>
</tr>
<tr>
<td></td>
<td></td>
<td>components/publisher</td>
<td>(softwareCreator/publisher), @name</td>
</tr>
<tr>
<td>Component Name</td>
<td>(3.1) PackageName:</td>
<td>metadata/component/</td>
<td>&lt;softwareIdentity&gt; @name</td>
</tr>
<tr>
<td></td>
<td></td>
<td>name</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>components/name</td>
<td></td>
</tr>
<tr>
<td>Version String</td>
<td>(3.3) PackageVersion:</td>
<td>components/version</td>
<td>&lt;softwareIdentity&gt; @version</td>
</tr>
<tr>
<td>Component Hash</td>
<td>(3.10) PackageChecksum:</td>
<td>components/hashes</td>
<td>&lt;Payload&gt;/../&lt;File&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>@([hash-algorithm]:hash</td>
</tr>
<tr>
<td></td>
<td>(3.9) PackageVerificationCode:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unique Identifier</td>
<td>(2.5) SPDX Document</td>
<td>serialNumber</td>
<td>&lt;softwareIdentity&gt; @tagID</td>
</tr>
<tr>
<td>Namespace</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(3.2) SPDXID:</td>
<td>components/bom-ref</td>
<td></td>
</tr>
<tr>
<td>Relationship</td>
<td>(7.1) Relationship:</td>
<td>Dependencies</td>
<td>&lt;Link&gt; @rel, @href</td>
</tr>
<tr>
<td></td>
<td>DESCRIBES</td>
<td>compositions</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CONTAINS</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Mapping baseline component information to existing formats
Figure 3: User graph with two supply chains
<table>
<thead>
<tr>
<th>Component Name</th>
<th>Supplier Name</th>
<th>Version String</th>
<th>Author</th>
<th>Hash</th>
<th>UID</th>
<th>Relationship</th>
<th>Relationship Assertion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Acme</td>
<td>1.1</td>
<td>Acme</td>
<td>0x123</td>
<td>234</td>
<td>Primary</td>
<td>Known</td>
</tr>
<tr>
<td></td>
<td>---</td>
<td>Browser</td>
<td>Bob</td>
<td>2.1</td>
<td>Bob</td>
<td>0x223</td>
<td>334</td>
</tr>
<tr>
<td></td>
<td></td>
<td>---</td>
<td>Compression Engine</td>
<td>Carol</td>
<td>3.1</td>
<td>Acme</td>
<td>0x323</td>
</tr>
<tr>
<td></td>
<td>---</td>
<td>Buffer</td>
<td>Bingo</td>
<td>2.2</td>
<td>Acme</td>
<td>0x423</td>
<td>534</td>
</tr>
</tbody>
</table>

Table 4: Conceptual SBOM table with upstream relationship assertions

<table>
<thead>
<tr>
<th>Component Name</th>
<th>Supplier Name</th>
<th>Version String</th>
<th>Author</th>
<th>Hash</th>
<th>UID</th>
<th>Relationship</th>
<th>Relationship Assertion</th>
</tr>
</thead>
<tbody>
<tr>
<td>NanoPhone</td>
<td>Nancy</td>
<td>v1254-a4</td>
<td>Nancy</td>
<td>0x523</td>
<td>237</td>
<td>Primary</td>
<td>Partial</td>
</tr>
<tr>
<td></td>
<td>---</td>
<td>OpenLibrary</td>
<td>Oscar</td>
<td>0.9.8s</td>
<td>Nancy</td>
<td>0xA23</td>
<td>394</td>
</tr>
<tr>
<td></td>
<td>---</td>
<td>Protocol</td>
<td>Paul</td>
<td>2012.11</td>
<td>Nancy</td>
<td>0xB53</td>
<td>934</td>
</tr>
</tbody>
</table>

Table 6: Conceptual SBOM table representation for Nancy’s NanoPhone
Summary SBOM Processes

• Define components, produce, maintain, provide SBOM
  • SBOM generation happens around build, package, deployment
• Seek, request, require SBOM from upstream suppliers
  • Sector-specific regulation, acquisition
• If upstream SBOM unavailable, make one up
• Choose existing formats and exchange mechanisms
When should an SBOM be used?

Source: NTIA's Survey of Existing SBOM Formats and Standards
Different Types of SBOMs

- Open Source Projects
- 3rd Party

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

upstream / supplier

- Source SBOM
- Build SBOM
- Binary SBOM

downstream / consumer

Open Source Projects

Customers
## Taxonomy 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>SBOM is automatically created as part of building a software artifact and contains information about the build</td>
</tr>
<tr>
<td></td>
<td>Analyze</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></td>
<td>Edit</td>
<td>A tool to assist a person manually entering or editing SBOM data</td>
</tr>
<tr>
<td>Consume</td>
<td>View</td>
<td>Be able to understand the contents in human readable form (e.g. 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 multiple SBOMs and clearly see the differences (e.g. comparing two versions of a piece of software)</td>
</tr>
<tr>
<td></td>
<td>Import</td>
<td>Be able to discover, retrieve, and import an SBOM into your system for further processing and analysis</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 SBOM and other data can be combined 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, transport, or other reference sources</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 (Analyze, Edit), Consume(View, Diff, Import), Transform(Translate, Merge, Tool Support)</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>
<tr>
<td>Location</td>
<td>Website: <a href="https://www.fossology.org/">https://www.fossology.org/</a> Source: <a href="https://github.com/fossology">https://github.com/fossology</a></td>
</tr>
<tr>
<td>Installation instructions</td>
<td><a href="https://www.fossology.org/get-started/">https://www.fossology.org/get-started/</a></td>
</tr>
<tr>
<td>How to use</td>
<td><a href="https://www.fossology.org/get-started/basic-workflow/">https://www.fossology.org/get-started/basic-workflow/</a></td>
</tr>
<tr>
<td>Versions Supported:</td>
<td>SPDX 2.1, SPDX 2.2</td>
</tr>
</tbody>
</table>
Collecting the Tools...

- Google docs for collecting tools in the three formats (open source and commercial offerings)
  - 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)

- Desire to move to neutral GitHub location to allow a more open process and wider set of visible reviews.
  - Anyone can nominate tool to be added to a list
  - Point to evidence of producing, consuming or transforming of SBOM documents to get tool on the list (this includes participating in Plugfest)
Translating between SBOM Formats & File Types

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

- **SPDX online tools**: ( SPDX (.spdx, .json, .yaml, .rdf, .xml, .xls) )
  - Demo at: [https://tools.spdx.org/app/](https://tools.spdx.org/app/)
  - Source code at: [https://github.com/spdx/spdx-online-tools](https://github.com/spdx/spdx-online-tools)

- **CycloneDX CLI**: ( CycloneDX (.xml, .json), SPDX(.spdx))
  - Source code at: [https://github.com/CycloneDX/cyclonedx-cli](https://github.com/CycloneDX/cyclonedx-cli)
Where to find more info on tools:

- **CycloneDX**: [https://cyclonedx.org/tool-center/](https://cyclonedx.org/tool-center/)
- **SPDX**: [https://spdx.dev/resources/tools/](https://spdx.dev/resources/tools/)

Possible Next Steps:

- Plugfests in 2022 (Consumers, ???)
- **Case studies** of organization adoption of tools & **reference** tooling workflows
SPDX: Overview

William Bartholomew (@iamwillbar)
Principal Security Strategist, Microsoft
Core Profile Lead, SPDX
Mission

The mission of SPDX is to develop and promote open standards for communicating software bill of material information (SBOM), including provenance, license, security, and other related information.
Background

😊 Born out of a need to exchange OSS component and license information

🎂 Recently celebrated its 10th birthday

🎉 SPDX 2.2 became ISO standard this year (ISO/IEC 5962:2021)

ℹ️ Open weekly working group and monthly general meetings
Features

Multiple formats (JSON, YAML, RDF/XML, Tag/Value)
Flexible for different use cases
Rich open source licensing expressions
Describe complex relationships
Cross-document references
Example

SPDXVersion: SPDX-2.2
DataLicense: CC0-1.0
SPDXID: SPDXRef-DOCUMENT
DocumentName: hello
DocumentNamespace: https://swinslow.net/spdx-examples/example1/hello-v3
Creator: Person: Steve Winslow (steve@swinslow.net)
Creator: Tool: github.com/spdx/tools-golang/builder
Creator: Tool: github.com/spdx/tools-golang/idsearcher
Created: 2021-08-26T01:46:00Z

PackageName: hello
SPDXID: SPDXRef-Package-hello
PackageDownloadLocation: git+https://github.com/swinslow/spdx-examples.git#example1/content
FilesAnalyzed: true
PackageVerificationCode: 9d20237bb72087e6f801b537b64d4fdccaf6f5ac0f2
PackageLicenseConcluded: GPL-3.0-or-later
PackageLicenseInfoFromFiles: GPL-3.0-or-later
PackageCopyrightText: NOASSERTION
FileName: /build/hello
SPDXID: SPDXRef-hello-binary
FileType: BINARY
FileChecksum: SHA1: 20291a81ef065ff801b537b64d4fdccaf6f5ac0f2
FileChecksum: SHA256: 83a33ff09648dbbf5cf5272baca88cf2b59fd81ac4cc6817b8699813ef368708e
FileChecksum: MD5: 08a12c966d776b64cc1eb41fd03c3c3d
LicenseConcluded: GPL-3.0-or-later
LicenseInfoInFile: NOASSERTION
FileCopyrightText: NOASSERTION

FileName: /src/hello.c
SPDXID: SPDXRef-hello-src
FileType: SOURCE
FileChecksum: SHA1: 20862a6d083910d07d9344029533ec644f6a6c6b20
FileChecksum: SHA256: b4e5ca56d1f9110ca94ed0bf4e6d9ac11c2186eb7cd95159c6fd650e8db5a823
FileChecksum: MD5: 935054fe899ca782e11003bbae5e166c
LicenseConcluded: GPL-3.0-or-later
LicenseInfoInFile: GPL-3.0-or-later
FileCopyrightText: Copyright Contributors to the spdx-examples project.

Relationship: SPDXRef-DOCUMENT DESCRIBES SPDXRef-Package-hello
Relationship: SPDXRef-hello-binary GENERATED_FROM SPDXRef-hello-src
SPDX 3.x

Split specification into profiles
- Core (Artifact, Relationship, Collection, IntegrityMethod, Identity, ...)
- Software (Package, File, Snippet, SBOM, ...)
- Licensing

Minimize required fields

New profiles
- Defects (Vulnerability, ...)
- Usage

Support scenarios beyond software
- Hardware
- Services
- Data
Next steps

- Learn https://spdx.dev/
- Participate https://spdx.dev/participate/
- Use https://spdx.dev/resources/tools/
CycloneDX
Software Bill of Materials Standard
Patrick Dwyer

- Co-Leader of OWASP CycloneDX
- Contributor to multiple SBOM related projects and tools
- OSS Maintainer
- Software Development Lead (Government)

@coderpatros
patrick.dwyer@owasp.org
Introducing CycloneDX

- Flagship OWASP standards project
- Lightweight, simplicity over complexity - easy to implement and adopt
- Optimized for highly automated processes
- Purpose built as a BOM format for cybersecurity use cases
- Designed in May 2017
- Initial release March 2018
- Yearly releases since
- Formal governance and standards process
- Recommended by multiple world government agencies
- Large and growing industry and vendor support
  - https://cyclonedx.org/about/supporters/
- Estimated to be in use at 100k organizations
Use Case Examples

A collection of common use cases achievable with CycloneDX along with concrete examples in XML and JSON.

https://cyclonedx.org/use-cases/
BOM Metadata

{
  "bomFormat": "CycloneDX",
  "specVersion": "1.3",
  "serialNumber": "urn:uuid:3e671687-395b-41f5-a30f-a58921a69b79",
  "version": 1,
  "metadata": {
    {
      "timestamp": "2020-04-13T20:39:00Z",
      "tools": [ ... ],
      "authors": [ ... ],
      "manufacture": { ... },
      "supplier": { ... },
      "component": { ... }
    }
  }
}
Component Inventory

```json
{
    "components": [
        {
            "type": "library",
            "group": "org.apache.logging.log4j",
            "name": "log4j2-core",
            "version": "2.14.1"
        }
    ]
}
```

Supports:
- Applications
- Libraries
- Frameworks
- Containers
- Operating systems
- Firmware
- Devices
- Files
- Services
Known vulnerabilities

```json
{
    ...
    "components": [
        {
            "type": "library",
            "group": "org.apache.logging.log4j",
            "name": "log4j2-core",
            "version": "2.14.1",
            "purl": "mvn:org.apache.logging.log4j/log4j-core@2.14.1",
            "swid": { ... }
        }
    ]
}
```
Integrity

{
...
"components": [
{
    "type": "library",
    "group": "org.apache.logging.log4j",
    "name": "log4j2-core",
    "version": "2.14.1",
    "hashes": [
        {"alg": "SHA3-512", "content": "..." }
    ]
}
]}
}
Authenticity

- XML Signature
- JSON Web Signature (JWS)
- JSON Signature Format (JSF)
- Digital signatures can be applied to a BOM or to an assembly within a BOM
- Signatures can be external to the BOM or enveloped (included within)
Component Pedigree

"pedigree": {
  "ancestors": [
    {
      "type": "library",
      "group": "org.apache.logging.log4j",
      "name": "log4j2-core",
      "version": "2.14.1"
    }
  ],
  "patches": [
    {
      "type": "backport",
      "diff": { "...", "resolves": [{ "type": "security", "id": "CVE-2021-44228", ... }] } } 
  ]
}
Provenance

- Component downloaded location
- Supplier
- Author
- Publisher
Composition

- Assemblies
- Dependency graph
- Completeness
  - complete
  - incomplete
  - first-party/third-party
  - unknown
and many, many more...
Tool Center

Community effort to establish a marketplace of free, open source, and proprietary tools and solutions that support CycloneDX.

https://cyclonedx.org/tool-center/
In development

- Improved hardware support
- “Vulnerability-Exploitability eXchange” format, aka VEX
- IETF URN namespace registration to deeplink between BOMs
- Schema hardening
- OWASP SBOM Maturity Model
- CycloneDX v1.4 due for release January 2022
Community Participation

- Website (introduction, use cases, tool center, and specification)
  - [https://cyclonedx.org/](https://cyclonedx.org/)
- GitHub
  - [https://github.com/CycloneDX](https://github.com/CycloneDX)
- Slack
  - [https://cyclonedx.org/slack](https://cyclonedx.org/slack)
  - [https://cyclonedx.org/slack/invite](https://cyclonedx.org/slack/invite)
- Mailing List
  - [https://cyclonedx.org/discussion](https://cyclonedx.org/discussion)
Thank You
Standing on Shoulders:
A Review of Playbooks from NTIA’s SBOM Multistakeholder Initiative

JC Herz
jc.herz@ionchannel.io
Overview: Strategic and Tactical Objectives

- SBOM file formats are established and in commercial use
- BUT: files are a starting point, not the end state
- Operationalizing SBOMs requires workflows to generate and act on the data
- Supplier playbook: Steps to create and provide
- Consumer playbook: Steps to metabolize and use
- Reality: Most suppliers are also consumers, and often vice versa
- Playbooks are technical and business process requirements - not endorsements of specific technological solutions or build/buy decisions.
Supplier Playbook

● SBOM Production: Generalized Process
  1. Identify software components included in a deliverable
  2. Acquire data about components used in a deliverable
  3. Import component data into a structured SBOM format.
  4. Validate SBOM to ensure format is valid and baseline attributes are present.

● Relevant Workflow Differentiators
  ○ Best-Practice vs. Non-Automated Engineering Processes
  ○ Build-Time vs. Post-Build SBOMs

● Deliverable (What’s in the Box): Applications, Containers, Systems
  ○ Ex: Operating System, Runtime Dependencies, Installers

● Requires Consensus: External Services

Consumer Playbook

- Acquisition of SBOM from a Supplier
  - Contractual procurement of a commercial product
  - Download of commercial closed-source product
  - Contractual procurement of professional services
  - Acquisition of open source software
  - Discovery processes as a device connects to a network

- SBOM Coverage for Software Systems
- Software Entity Resolution
- Third Party Processes and Platforms
- Ongoing Monitoring
- Ideally we should be able to assure critical software as well as we can assure a steak.

Vulnerability Exploitability Exchange (VEX)

Jens Wiesner
Head of Section
German Federal Office for Information Security (BSI)
We will know about more potential vulnerabilities with SBOM
Not all vulnerabilities are exploitable
Not all vulnerabilities are exploitable

- Component not present
- Affected code not loaded
- In-line mitigations exist
- Affected code not in path
- Attacker can’t touch affected code
Zabbix NOT AFFECTED by the Log4j exploit

A newly revealed vulnerability impacting Apache Log4j 2 versions 2.0 to 2.14.1 was disclosed on GitHub on 9 December 2021 and registered as CVE-2021-44228 with the highest severity rating. Log4j is an open-source Java-based logging utility widely used by enterprise applications and cloud services. By utilizing this vulnerability, a remote attacker could take control of the affected system.

Zabbix is aware of this vulnerability, has completed verification, and can conclude that the only product where we use Java is Zabbix Java Gateway, which does not utilize the log4j library, thereby is not impacted by this vulnerability.

For customers, who use the log4j library with other Java applications, here are some proactive measures, which they can take to reduce the risk posed by CVE-2021-44228:

We need a way to communicate that a product is not affected
Do I need to do anything?

You're good.

"exploitable"

"affected"

"not affected"
**“exploitable”**

Actions are recommended to remediate or address this vulnerability. This could include: learning more about the vulnerability and context, and/or making a risk-based decision to patch or apply defense-in-depth measures.

**“affected”**

**“not affected”**

No remediation is required regarding this vulnerability. This could be because the code referenced in the vulnerability is not present, not exposed, compensating controls exist, or other factors.
Required fields for a VEX

<table>
<thead>
<tr>
<th>Metadata</th>
<th>(author, id, timestamp)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product id</td>
<td></td>
</tr>
<tr>
<td>Vulnerability ID</td>
<td></td>
</tr>
<tr>
<td>Vuln details</td>
<td></td>
</tr>
<tr>
<td><strong>Product Status</strong></td>
<td></td>
</tr>
<tr>
<td>Action statement /</td>
<td></td>
</tr>
<tr>
<td>Impact statement</td>
<td></td>
</tr>
</tbody>
</table>

Implementing VEX in Common Security Advisory Framework (CSAF)
Common Security Advisory Framework

- Original purpose: automate security advisories to support search and evaluation
- CSAF 2.0
  - JSON format
  - Machine-readable
  - Build with automation in mind
- Standardization through CSAF TC at OASIS Open
- Successor of CSAF CVRF 1.2
- VEX implemented as a profile in CSAF
- VEX is parallel to SBOM (not necessarily in the SBOM)
CSAF Information: [csaf.io](csaf.io)
More info on CSAF: [www.bsi.bund.de/EN/Topics/Industry_CI/ICS/Tools/CSAF/csaf_node.html](www.bsi.bund.de/EN/Topics/Industry_CI/ICS/Tools/CSAF/csaf_node.html)
CSAF Editor & Examples: [secvisogram.github.io/](secvisogram.github.io/)
Join the VEX working group: [sbom@cisa.dhs.gov](sbom@cisa.dhs.gov)
SBOM Proof of Concept

HEALTHCARE
Let’s frame the issue

Standards & formats

We’ll identify use cases
A Healthcare SBOM
Proof of Concept is Born

Let’s try SBOMs. Now.

Manufacturers
SBOM producers

Providers
SBOM consumers
# A Healthcare Proof of Concept: Crawl, Walk, Run

how we set out to prove the viability of generating standardized SBOMs

<table>
<thead>
<tr>
<th>Phase</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase I</td>
<td>Investigate 2018-2019: Custom tools/manual processes to generate SBOMs in standard formats</td>
</tr>
<tr>
<td>Phase II</td>
<td>Iterate 2020-2021: Expand scope and modify processes based upon findings</td>
</tr>
<tr>
<td>Phase III</td>
<td>Integrate 2022: Automate with new tools and processes in existing tool chains and systems</td>
</tr>
</tbody>
</table>
# Healthcare Proof of Concept

## Goals and accomplishments by phase

<table>
<thead>
<tr>
<th>Phase</th>
<th>Investigate</th>
<th>Iterate</th>
<th>Integrate</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Phase I</strong></td>
<td>2018-2019</td>
<td>2020-2021</td>
<td>2022</td>
</tr>
</tbody>
</table>
| **Goals** | Exercised primary use cases in creating & ingesting SBOMs for risk mgmt. ✔  
Proved the actionable value of component transparency to the consumer ✔ | More participants, more use cases, more devices, more data, more tools ✔  
Proved the viability of standard formats, data, tools; explored context info ✔ | Drive adoption, expanded participation; real-world scenarios & data  
Automate SBOM sharing; prove value of context (VEX, support lifetimes) |
| **Notes** | | | During Covid! |
What has been happening to help?

Starting again in February 2022

https://inl.gov/sbom-poc/
Cooking Classes

• Cooking Class on Making an SBOM, Sept. 22, 2021 - https://youtu.be/Tk4v1IrSNSA
• Cooking Class on Open Source, Oct. 6, 2021 - https://youtu.be/5D0P84ayGpg
• Cooking Class on VEX, Oct 20, 2021 - https://youtu.be/KjMHxeHYgIQ
• Cooking Class on Preparing to Use SBOM’s, Nov. 3, 2021 - https://youtu.be/Tqkdb3XvR08

https://inl.gov/sbom-poc/
4.2.6 Inventory and Management of Software Assets on Vehicles

[G.10] Manufacturers should maintain a database of operational software components\textsuperscript{19,20} used in each automotive ECU, each assembled vehicle, and a history log of version updates applied over the vehicle’s lifetime.

[G.11] Manufacturers should track sufficient details related to software components,\textsuperscript{21} such that when a newly identified vulnerability is identified related to an open source or off-the-shelf software,\textsuperscript{22} manufacturers can quickly identify what ECUs and specific vehicles would be affected by it.

4.2.7 Penetration Testing and Documentation

Draft 2020 Update

that support the intended protections.
May 2021 - Executive Order 14028 - “Improving the Nation’s Cybersecurity”
AutoISAC SBOM Working Group - History

**AutoISAC Phase 1 – Mar-Jul 2019**

- **Sponsor:** Analyst WG
- **Goal:** Ensure NTIA SBOM considers automotive industry issues and opinions
- **Team:** 10 members (includes 3 OEMs)
- **Objective:** Publish concerns to NTIA and advocate for the auto industry

**AutoISAC Phase 2 – Nov 2020 – Dec 2021**

- **Sponsor:** Supplier Affinity Group
- **Goal:** Agree on best practices among suppliers and propose solution to OEMs
- **Team:** 17 members (1 OEM)
- **Objectives:**
  - Unified supplier voice on SBOM adoption to OEMs
  - Align with NTIA
  - Practical approach with input from OEMs
  - Best Practice published in 2021

**NTIA – July 2018 – November 2021**

**Hitachi – November 2018 – Dec 2021**
Preview: Best Practice Guide Proposal

WILL INCLUDE

- TLP AMBER distribution (for now)
- Substantial overlap with NTIA guidance
- Customizations for automotive
- Mapping to automotive product lifecycle
- Format and operational recommendations
- Sharing discussion
- Vendor-neutral tool list
- Bibliography, training, and reference docs

WILL NOT INCLUDE

- Mandatory rules – all points will be recommendations
- Usurpation of supplier contracts or requirements
- Static guidance – revisions expected during Phase 3 and ongoing
Next Steps

1. Finalize Best Practice Draft Proposal
2. Board of Directors approval
3. Phase 3 ( Likely ) – active exercise – details under discussion
4. Future Possibilities ( not decided )
   Limited production pilot exercise
   Training program
   Automation and tool trials
   DHS/CISA program ( NTIA successor )
   Supply chain integrity exercise
   Vulnerability management use case and exercise
   Addition of Vulnerability/Exploitability eXchange ( VEX ) automation