Managing software supply chain risk
Role of a comprehensive SBOM

Tim Mackey
Head of Software Supply Chain Risk Strategy
Software & Supply Chain Assurance (SSCA) Forum May 2023
#whoami

- Joined Synopsys in 2017 as part of the Black Duck acquisition

- Currently Head of Software Supply Chain Risk Strategy
  - Co-chair of DHS CISA ICT SCRM Task Force on SwA

- Previously Head of Technical Marketing and Partner Product Marketing

- Worked 13 years at Citrix
  - Virtualization and Cloud lead within Citrix Open Source Business Office
  - Dotted line to Citrix CSO with product security responsibility
  - Thought leadership in virtualization and containerization efforts
Software operators assume risk from software supply chains

**Visibility into Software Supply Chain Risk**

**Software producers**

*How do we **build trust** with our customers and end users?*

**Software operators**

*How do we **maintain visibility** and control of risks?*

**Software Supply Chain Complexity and Risk**
Essentials of Software Supply Chain Trust

Trust objectives

- Build and operate secure applications
- Attest to development security efforts
- Adhere to software license obligations
- Verify integrity of software

**Trust** is the result of supply chain **integrity**.

**Integrity** is the result of consistently executed **processes**.

Effective **processes** are the result of **collaboration** between software producers and operators.
Implementation decisions in code your teams write
(proprietary code)

+ 

Implementation, testing and release decisions in code you consume
(open source, container base images and 3rd party vendor libraries)

+ 

Implementation, testing and deployment decisions in cloud services
(API usage and dataflows)
Cyber risk assessments within supply chains are still rare

Percentage of UK organizations that have carried out work to formally review the potential cyber security risks presented by the following groups of suppliers

Unchanged from 2022 Survey

Up 11 points since 2022 Survey

Down 8 points since 2022 Survey

Source: Cyber Security Breaches Survey 2023 – UK Department for Science, Innovation and Technology
Software supply chain risk efforts are often siloed

And risk is more than just an unpatched vulnerability

Applies to all software that is:
- Bought with hardware
- Bought as software
- Built internally
- Downloaded
- Contracted
- Modified
- Updated or patched
SBOMs are now key to software risk management
You can’t talk software supply chain risk without SBOMs

Two primary SBOM standards to be aware of – and expect both to be supported

- SPDX was created in 2010
- Is part of Linux Foundation Open Compliance Program
- Synopsys is a supporting partner
- Version 2.2.1 is international standard ISO/IEC 5962:2021
- Latest version: 2.3 - released Nov 2022
- Version 3.0 has anticipated release of summer 2023
- Six core specification contributors

- CycloneDX was created in 2017
- Is part of OWASP and intended for use with OWASP Dependency-Track
- Synopsys is a supporting organization
- Latest version: 1.4 - released January 2022
- One core specification contributor

Both formats meet the NTIA minimum requirements
SBOMs serve different markets – with different requirements

Medical devices – FDA Reg 524B* and IMDRF/N73FINAL.2023

US Government Software Procurement – Executive Order 14028 and GSA Memo MV-23-02

Automotive – ISO 21434, UNR 155 and NHTSA Critical Infrastructure – IEC 62443

Same SBOM formats, different minimum fields, different supporting documentation

Requirements are global and not US centric

Note: FDA regulation 524B requires SBOM and vulnerability response plans starting March 29, 2023
Examples of SBOM definitions – more than just NTIA

### FDA Requirements for SBOM[^1]

- NTIA Minimum SBOM Fields[^2]
- NHSTA Recommended Software Inventory Management[^3]

<table>
<thead>
<tr>
<th>Data Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supplier Name</td>
<td>The name of an entity that creates, defines, and identifies components.</td>
</tr>
<tr>
<td>Component Name</td>
<td>Designation assigned to a unit of software defined by the original supplier.</td>
</tr>
<tr>
<td>Version of the Component</td>
<td>Identifier used by the supplier to specify a change in software from a previously identified version.</td>
</tr>
<tr>
<td>Other Unique Identifiers</td>
<td>Other identifiers that are used to identify a component, or serve as a look-up key for relevant database.</td>
</tr>
<tr>
<td>Dependency Relationship</td>
<td>Characterizing the relationship that an upstream component X is included in software Y.</td>
</tr>
<tr>
<td>Author of SBOM Data</td>
<td>The name of the entity that creates the SBOM data for this component.</td>
</tr>
<tr>
<td>Timestamp</td>
<td>Record of the date and time of the SBOM data assembly.</td>
</tr>
</tbody>
</table>

[^1]: [FDA – Cybersecurity in Medical Devices: Quality System Considerations and Content of Premarket Submissions (Draft Apr 2022)](https://www.fda.gov)


Sources: [1] FDA – Cybersecurity in Medical Devices: Quality System Considerations and Content of Premarket Submissions (Draft Apr 2022)


Example: SBOM in automotive appears in multiple guidelines

Security standards and increased use of open-source libraries create a new rule book

**ISO/SAE 21434** – Clause 6, Clause 7, RQ-08-07, RC-10-12
- Component reuse and usage of off-the-shelf components \( (SBOM) \) shall include an understanding of the suppliers’ cybersecurity activities and vulnerability management practices

**UNR 155** – Section 5.1.1
- Vehicle manufacturer has taken appropriate measures to collect and verify the information required through the [software] supply chain \( (SBOM) \) so as to demonstrate supplier related risks are identified and managed

**NHTSA Cybersecurity Best Practices 2022** – Section G.10
- Suppliers and vehicle manufacturers should maintain a database of their operational hardware and software components \( (SBOM) \) used in each automotive ECU

**RQ-08-01** Cybersecurity monitoring \( (SBOM) \)
Result: SBOM responsibilities increase in the value chain

Value chain tiers add software library usage, risk and map in safety documentation
Comprehensive SBOMs identify the origin of all components

Created by software producers for the benefit of software operator/consumer

WHAT IT IS
List of application dependencies (ingredients)

WHAT IT IS NOT
Vulnerability feed, operational risk, or implementation insights
Robust SCA is a requirement for all SBOM management

- **Inventory**: Identify and track all libraries in apps and containers
- **Security**: Find and fix known disclosed vulnerabilities in development and production
- **License Compliance**: Verify and comply with open source license terms and conditions
- **Governance and Policy**: Integrate and automate open source risk policy enforcement end-to-end
Automate comprehensive SBOM generation in the SDLC

Accurate and complete SBOM generation relies upon multiple analysis techniques

- Code
- Build
- Test
- Operate

Dependency analysis

Signature analysis

Binary analysis

Snippet analysis

**OSS BoM – continuously monitored**
### Two key challenges for comprehensive SBOMs

#### Completeness
- Can the component be identified?
  - E.g. ADD, COPY, RUN in containers
- Does knowledge source contain the component?
  - E.g. missing forges or incomplete understanding of FOSS
- Does the knowledge source align to the applications' lifespan
  - E.g. Will it retain information for legacy items?
- How are internal components and 3rd party commercial non-COTS components handled?

#### Accuracy
- Was the SBOM created using tooling?
  - Manually created SBOMs are prone to being outdated
- Does SBOM adhere to schema for chosen format?
- Does tooling support precise version identification?
- Does SBOM map to known software release?
  - E.g. integrity check for release vs SBOM
- Was SBOM created from merged SBOMs?
  - Was source accuracy validated?
Knowing where an SBOM is created in the SDLC matters

SBOM contents can vary, and data fidelity will vary accordingly

Source: Terms used are consistent with CISA SBOM Types April 2023
Comprehensive SBOMs enable supply chain risk analysis

Mapping risk elements to SBOMs is the key to extensibility and visibility

Rather than treat the SBOM as a single model for all assurance and software supply chain data, a linkable, modular approach is encouraged to maximize the potential for flexibility and adoption.

Linkability enables SBOM data to be easily mapped to other important supply chain data, while a modular architecture supports extensibility for more use cases as software supply chain transparency and management data and tools mature.

Source: NTIA - The Minimum Elements For a Software Bill of Materials (SBOM)
Example: Disclosing unpatched CVEs present at release

Use NIST Vulnerability Disclosure Report (VDR) process mapped to SBOM data

- Component 1
- Component 2
- Apache Log4j 1.2.17
- Component 3
- Component 4

CVE-2019-17571
CVE-2022-23307
CVE-2020-9493
CVE-2023-26464
CVE-2021-4104

Not vulnerable because: “CVE-2021-4104 is not impacted because there is no usage of the JMSAppender class”

CVE mitigation information provided as part of vulnerability report
Sidebar: Vulnerability Exploitability eXchange (VEX) process

Not directly connected with SBOMs, but part of the SBOM ecosystem

- **Goal**: Provide transparency of vulnerability data to software operators
- **Core requirement**: Identify in automation or tooling what isn’t impacted by a CVE
- **Mature implementation**: Common Security Advisory Framework (CSAF) 2.0
- **Minimum requirements published**: April 21, 2023 by DHS CISA*

Example VEX statement

```
"product_status": {
"fixed": [
"8Base-RHOSE-4.11:openshift4/ose-etcd:v4.11.0-202301041324.p0.gc50e9aa.assembly.stream",
],
"known_not_affected": [
"8Base-RHOSE-4.11:openshift4/cloud-network-config-controller-rhel8:v4.11.0-202301051554.p0.g5dd318b.assembly.stream",
"8Base-RHOSE-4.11:openshift4/network-tools-rhel8:v4.11.0-202301070325.p0.g4e87286.assembly.stream",
"8Base-RHOSE-4.11:openshift4/ose-baremetal-installer-rhel8:v4.11.0-202301042055.p0.gf746e45.assembly.stream",
"8Base-RHOSE-4.11:openshift4/ose-cluster-baremetal-operator-rhel8:v4.11.0-202301091615.p0.g1f1ea53.assembly.stream",
[, ...full list truncated...]
]
}
```

*https://www.cisa.gov/resources-tools/resources/minimum-requirements-vulnerability-exploitability-exchange-vex
### Twelve essential elements for assuring trust in software supply chain

<table>
<thead>
<tr>
<th></th>
<th>Asset inventory</th>
<th>Know what software you have (and perhaps don't want!)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>SBOM</td>
<td>Produce a Software Bill of Materials (SBOM) on demand</td>
</tr>
<tr>
<td></td>
<td>Provenance of software</td>
<td>Know the origin of your software</td>
</tr>
<tr>
<td>4</td>
<td>Secure dev environment</td>
<td>Secure the development pipeline and artifacts</td>
</tr>
<tr>
<td>5</td>
<td>Integrity attestation</td>
<td>Attest to integrity of a software artifact (e.g. component, zip file, container, API connection, database connection)</td>
</tr>
<tr>
<td>6</td>
<td>Quality</td>
<td>Identify bugs and reliability issues</td>
</tr>
<tr>
<td>7</td>
<td>Security</td>
<td>Identify security issues</td>
</tr>
<tr>
<td>8</td>
<td>Regulatory noncompliance</td>
<td>Determine noncompliance with regulations and standards</td>
</tr>
<tr>
<td>9</td>
<td>Licensing noncompliance</td>
<td>Ensure software is legally licensed</td>
</tr>
<tr>
<td>10</td>
<td>Unexpected functionality</td>
<td>Identify malware and other hidden or unexpected functions</td>
</tr>
<tr>
<td>11</td>
<td>Policy compliance</td>
<td>Show evidence that security policies are followed</td>
</tr>
<tr>
<td>12</td>
<td>Risk management reporting</td>
<td>Measure and communicate supply chain risks</td>
</tr>
</tbody>
</table>

Importance of each element depends on role and responsibility
### Twelve essential elements for assuring trust in software supply chain

<table>
<thead>
<tr>
<th></th>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Asset inventory</td>
<td>Know what software you have (and perhaps don’t want!)</td>
</tr>
<tr>
<td>2</td>
<td>SBOM</td>
<td>Produce a Software Bill of Materials (SBOM) on demand</td>
</tr>
<tr>
<td>3</td>
<td>Provenance of software</td>
<td>Know the origin of your software</td>
</tr>
<tr>
<td>4</td>
<td>Secure dev environment</td>
<td>Secure the development pipeline and artifacts</td>
</tr>
<tr>
<td>5</td>
<td>Integrity attestation</td>
<td>Attest to integrity of a software artifact (e.g. component, zip file, container, API connection, database connection)</td>
</tr>
<tr>
<td>6</td>
<td>Quality</td>
<td>Identify bugs and reliability issues</td>
</tr>
<tr>
<td>7</td>
<td>Security</td>
<td>Identify security issues</td>
</tr>
<tr>
<td>8</td>
<td>Regulatory noncompliance</td>
<td>Determine noncompliance with regulations and standards</td>
</tr>
<tr>
<td>9</td>
<td>Licensing noncompliance</td>
<td>Ensure software is legally licensed</td>
</tr>
<tr>
<td>10</td>
<td>Unexpected functionality</td>
<td>Identify malware and other hidden or unexpected functions</td>
</tr>
<tr>
<td>11</td>
<td>Policy compliance</td>
<td>Show evidence that security policies are followed</td>
</tr>
<tr>
<td>12</td>
<td>Risk management reporting</td>
<td>Measure and communicate supply chain risks</td>
</tr>
</tbody>
</table>

### Secure SDLC

- **Inventory & Operations**
- **Secure SDLC**
- **Verification, Compliance, Attestation**

A Secure Software Development Lifecycle is a necessary part of all software supply chains.
Crystal ball:
What an SBOM focused world might look like
Scenario: Operationalized SBOMs

Creating a continuum scaling SCA to meet InfoSec and product lifecycle requirements

**SCA**
- Is an AppSec tool
- Scans code
- Creates component BOM
- Maps BOM to vulnerabilities
- Generates OSS vulnerability disclosure records
- Publishes SBOM from vendor BOM
- Monitors for changes in BOM data
- Notifies developers upon new vulnerability
- Addresses AppSec persona
- Addresses IP compliance persona

**SBOM Management**
- Aggregates SBOM information from multiple providers, scans nothing
- Subscribes to OSS and commercial vulnerability feeds
- Generates vulnerability disclosure records for all components
- Implements continuous monitoring for new SSCRDM threats
- Notifies life cycle management teams upon policy violation (e.g. new OSS vulnerability)
- Publishes aggregate SBOM

**Lifecycle management**
- For IT organizations
  - Addresses IT requirements
  - Acts upon threat information
  - Creates post incident reports
- For DevOps organizations
  - Addresses product lifecycle team
  - Acts upon supply chain changes
  - Acts upon CVE disclosures
  - Creates product advisories

Operational SBOM breaks the AppSec barrier and scales SCA efforts beyond AppSec use cases by aggregating multiple risk elements and combining SBOMs from independent suppliers

Operational SBOM breaks the life cycle barrier by providing actionable information to ITSM/ALM/PLM tooling allowing the existing tooling to more efficiently react to changing external threats
Scenario: Find the “Sleeping Beast” in “stable code”

- 89% contained components that were more than 4 years out-of-date
- 91% contained components that weren’t the current version
- 91% contained components that had received no development activity in the last 2 years
- 88% contained components with no activity in the last 2 years and contained components that weren’t the latest version

- 91% of codebases contained components that had no new development in the past two years
- 89% of codebases contained open source more than four years out-of-date

Source: 2023 Synopsys Open Source Security and Risk Analysis report
Scenario: Identify if patch processes allow malicious code in?

Figure 1. Malicious package uploads to the npm code repository showed a 41% increase in 2022 over 2021, when researchers detected 4,940 packages. And the 2022 numbers represent more than a 9,000% increase over 2020, when researchers detected just 75 malicious npm packages.

Source: ReversingLabs
Scenario: Functional Safety and Traceability

SBOM enables software analysis beyond vulnerability management

Source: SPDX community
Scenario: MITRE System of Trust Framework

TRUST ASPECTS

SUPPLY RISKS

RISK CATEGORIES
- Supply Malicious Taint
- Supply Counterfeit
- Supply Hygiene Risks

SUPPLIER RISKS

RISK CATEGORIES
- Supplier Financial Stability Risks
- Supplier Organizational Security Risks
- Supplier Susceptibility
- Supplier Quality Culture Risks
- Supplier Organizational Effectiveness Risks
- Supplier Ethical Risks
- Supplier External Influences

SERVICE RISKS

RISK CATEGORIES
- Service Quality Risks
- Service Reliability Risks
- Service Security Risks
- Service Integrity Risks

Source: MITRE System of Trust
Core takeaways

• SBOMs are becoming a requirement, but you need a robust SCA solution to create them
  – Single SCA analysis technique rarely identifies all software

• Robust comprehensive SBOMs include all code: internal, 3rd party and open source
  – Advanced SBOM scenarios rely on deeper analysis of software risk from all software

• SBOM workflows are mostly focused on vulnerability management today
  – SCA vendors solved the problem of OSS vulnerability and license compliance years ago

• Products don’t have one single SBOM
  – The type of SBOM and how it was generated are key pieces of metadata
  – Cyber-physical devices may have many vendors requiring aggregation of SBOM data

• Having an SBOM doesn’t directly solve problems, but it does enable solutions
  – Push vendors to use SBOMs to solve critical problems not solved by SCA
Thank You