Software Bill of Materials
Progress towards Transparency in the SW Supply Chain

Allan Friedman, PhD
Director of Cybersecurity Initiatives, National Telecommunications and Information Administration, US Department of Commerce
afriedman@ntia.gov  @allanfriedman
Paying Attention vs. Checking email

Tracking and communicating third party components in software and IoT with a “software bill of materials” can:

• Improve and communicate secure development practices
• Help enterprise customers protect themselves
• Foster better markets for secure products

The US Department of Commerce has convened an open and consensus-driven multistakeholder process to develop a shared vision around SBOM and software transparency.

Stakeholders have drafted documents reviewing the what, the why, and the how.
If you have a 2013 Mercedes S-class you have libtiff, netcat, and libpcap, pre-installed.
We understand the role of a list of ingredients.
Analogies
In the manufacturing world, we track parts and components used in assembly to understand the manufacturing and maintenance process.
Analogies
No longer just an “emerging” risk
Software Supply Chain Attacks

1. Study found credentials online affecting publishing access to 14% of npm repository. +79,000 packages.
   Malicious npm Packages “typosquatting” (40 packages for 2 weeks. Collecting env including npm publishing credentials).

2. 10 Malicious Python packages
   Basic info collected and sent to Chinese IP address.

3. Blog: "I’m harvesting credit card numbers and passwords from your site. Here’s how."

4. Golang go-bindata github id deleted and reclaimed.

5. Conventional-changelog compromised and turned into a Monero miner.

   Unauthorized publishing of mailparser.

7. ssh-decorator Python Module stealing private ssh keys.

8. Gentoo Linux Repository Compromised.

9. Malicious Eslint discovered to be stealing npm credentials.

10. Homebrew repository compromised.

11. npm event-stream attack on CoPay.
300+ Backdoored Github Libraries
300+ Backdoored Github Libraries
Why don’t we do this today?
Enter: Your Friends, the Feds
Open, transparent, consensus-based processes that bring together diverse stakeholders can catalyze real progress across the ecosystem.
THIS MAN WAS
TALKED TO DEATH.
The problem to be solved
Modern software systems involve increasingly complex and dynamic supply chains.
Modern software systems involve increasingly complex and dynamic supply chains.

Lack of systemic transparency into the composition and functionality of these systems contributes substantially to cybersecurity risk as well as the costs of development, procurement, and maintenance.
Modern software systems involve increasingly complex and dynamic supply chains.

Lack of systemic transparency into the composition and functionality of these systems contributes substantially to cybersecurity risk as well as the costs of development, procurement, and maintenance.

In our increasingly interconnected world, risk and cost impact not only individuals and organizations directly but also collective goods like public safety and national security.
How a transparency solution can help

• Enhancing the identification of vulnerable systems and the root cause of incidents
How a transparency solution can help

• Enhancing the identification of vulnerable systems and the root cause of incidents
• Identifying suspicious or counterfeit software components
How a transparency solution can help

- Enhancing the identification of vulnerable systems and the root cause of incidents
- Identifying suspicious or counterfeit software components
- Reducing unplanned and unproductive work
How a transparency solution can help

- Enhancing the identification of vulnerable systems and the root cause of incidents
- Identifying suspicious or counterfeit software components
- Reducing unplanned and unproductive work
- Supporting more informed market differentiation and component selection
How a transparency solution can help

- Enhancing the identification of vulnerable systems and the root cause of incidents
- Identifying suspicious or counterfeit software components
- Reducing unplanned and unproductive work
- Supporting more informed market differentiation and component selection
- Reducing duplication of effort by standardizing formats across multiple sectors
• Harmonization
• Amplification & routinization
• Extensions & innovation
Making progress

• Clear appreciation across sectors on the potential value of transparency
  – The broad scope of the problem
  – Machine-readability of the solution
  – Focus on a minimum viable solution with extensions.
What is an SBOM?
The “minimum viable” SBOM

- Identity of Component
  - (Sufficient uniqueness)
- Relationship between components
- Extensions
Naming is Hard
Feature Support
SBOM as a graph
Being Clear about Opacity
Data about data

- I built this set of SBOM data
  vs
- This is SBOM data from someone else.
Why should we use an SBOM?
A supply chain perspective
A supply chain perspective

- Supplier selection
- Supply selection
- Supply vigilance
Capturing Stories

Each of these offers unique perspectives on the current and potential value of transparency.

We would love to have your perspective!
A supply chain perspective

- Writer/Maker
- Acquirer/Purchaser
- Operator/Maintainer
How do we SBOM?
Recall...
FORTUNATELY, WE HAVE SOME EXISTING TOOLS THAT WE CAN USE FOR SBOM DATA
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—all committed to creating a standard for software package data exchange formats.
SPDX Example

# Document Header
SPDXVersion: SPDX-2.1
DataLicense: CC0-1.0
SPDXID: SBOMDOCUMENT
DocumentName: SBOM-Proof-of-concept
DocumentNamespace: http://example.com
Created: 2018-12-18T22:11:34Z
CreatorComment: <text> This document was created as a proof-of-concept </text>

# Packages
PackageName: alsa-conf
SPDXID: yocto/alsa-conf@1.1.0
PackageVersion: 1.1.0
PackageDownloadLocation: NOASSERTION
FilesAnalyzed: false
PackageLicenseConcluded: NOASSERTION
PackageLicenseDeclared: NOASSERTION
PackageCopyrightText: NOASSERTION

PackageName: alsa-conf-base
SPDXID: yocto/alsa-conf-base@1.1.0
PackageVersion: 1.1.0
PackageDownloadLocation: NOASSERTION
FilesAnalyzed: false
PackageLicenseConcluded: NOASSERTION
PackageLicenseDeclared: NOASSERTION
PackageCopyrightText: NOASSERTION

PackageName: alsa-lib
SPDXID: yocto/alsa-lib@1.1.0
PackageVersion: 1.1.0
PackageDownloadLocation: NOASSERTION
FilesAnalyzed: false
PackageLicenseConcluded: NOASSERTION
PackageLicenseDeclared: NOASSERTION
PackageCopyrightText: NOASSERTION

Relationship: yocto/libasound2@1.1.0 PACKAGE_OF yocto/alsa-lib@1.1.0
Relationship: yocto/libc6@2.23.0 PACKAGE_OF yocto/alsa-lib@1.1.0

https://github.com/spdx/spdx-spec
Software Identification (SWID)

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.
SWID tag example

<SoftwareIdentity name="alsa-conf" tagId="yocto/alsa-conf@1.1.0" version="1.1.0"/>
<SoftwareIdentity name="alsa-conf-base" tagId="yocto/alsa-conf-base@1.1.0" version="1.1.0"/>
<SoftwareIdentity name="alsa-lib" tagId="yocto/alsa-lib2@1.1.0" version="1.1.0">
  <Link href="swid:yocto/libasound2@1.1.0" rel="requires"/>
  <Link href="swid:yocto/libc6@2.23.0" rel="requires"/>
</SoftwareIdentity>
...

...
WE HAVE IDENTIFIED THE COMMON ELEMENTS.
A ‘BILINGUAL’ ECOSYSTEM DOES NOT OFFER TOO MANY CHALLENGES

Rather than pick a winner, we will build out guidance to support both formats with effective interoperability.
Related efforts in the ecosystem

- Software Heritage Index
- Package URL (Purl)
- OpenChain
- CPE
Healthcare Proof of Concept
Open questions to figure out together
Obstacles to obtaining SBOM data?
Mechanisms of sharing SBOM data?

Vendors

Customers

Transparency
Vulnerability vs. Exploitability
Next steps

• Drafts of “minimum viable” by late June for feedback.
• After “minimum viable”
  – Extensions of data for use cases
  – Tooling
  – Awareness and Adoption
To recap...

- Tracking third party components can help understand and address a wide range of risks across the entire ecosystem.
- An ongoing, open process convened by NTIA is bringing together experts to address:
  - What a Software Bill of Materials is
  - Why it can help across the supply chain
  - How we can implement it
- Next steps will focus on tooling and extensions.
- Get involved in the NTIA process!
  - Contact afriedman@ntia.gov
  - @allanfriedman
Thank You!

afriedman@ntia.doc.gov