

Framing Software Supply Chain Transparency



Éamonn Ó Muirí

<https://flic.kr/p/46dsiz>

<https://creativecommons.org/licenses/by/2.0/legalcode>

Framing Working Group

Managed with love and patience by co chairs Michelle Jump and Art Manion

Meeting almost weekly since July 2018

- Fridays at 1400 EDT
- <https://lists.sei.cmu.edu/mailman/listinfo/ntia-sbom-framing>

Framing concepts that apply to the entire multi-stakeholder process



Michelle Jump

MedSec LLC

Global Regulatory Advisor -
Medical Device
Cybersecurity

MichelleJump@medsec.com



Art Manion

Software Engineering
Institute

CERT Coordination Center

Principle Engineer “/”
Technical Manager

amanion@cert.org



What is an SBOM?

Framing Software Component
Transparency: Establishing a Common
Software Bill of Material (SBOM)

<https://tinyurl.com/y7s8ab3t>

“An SBOM is effectively a nested inventory, a list of ingredients that make up software components.”

Partial Table of Contents

2 What is an SBOM?

2.2 Baseline Component Information

2.4 Component Relationships

4 SBOM Processes

4.1 SBOM Creation: How

4.2 SBOM Creation: When

4.3 SBOM Exchange

4.4 Network Rules

4.6 Applications of SBOMs

5 Terminology

Problems

We don't know what software is in our software

Greater transparency is necessary but not sufficient

- Affected by vulnerabilities in upstream dependencies?
- License compliance with upstream dependencies
- Confidence in the integrity of upstream dependencies
- Basic supply chain quality, hygiene, cost



SBOM Design

Terminology

- Supplier, author, consumer, attribute, component

Information model

- Baseline component information
- Relationships between components

Process model

- How and when to create SBOM
- Network rules
 - Supplier defines and identifies component



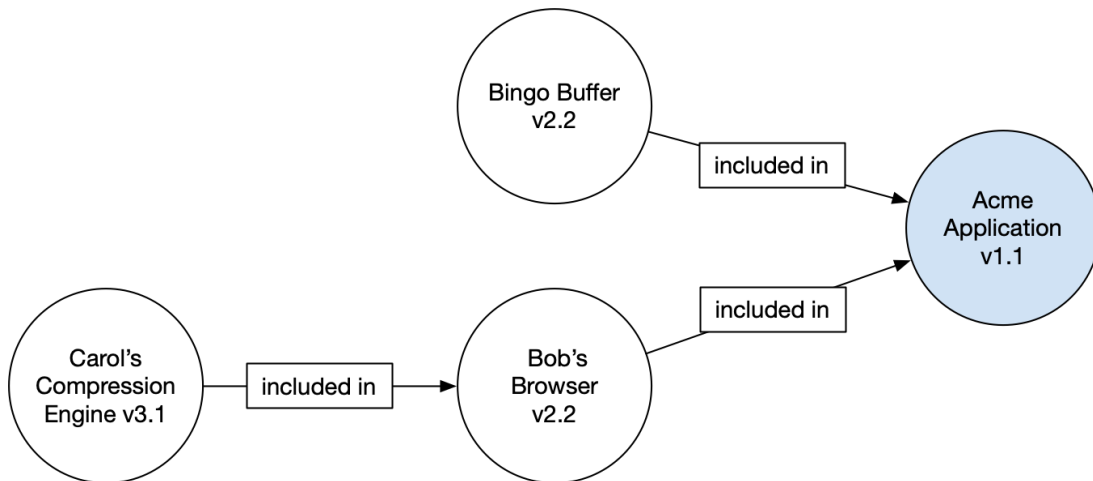
Baseline component information

Author name	Author of SBOM
Supplier name	Supplier of component Vendor, manufacturer, developer, maintainer
Component name	Supplier (or author) decides
Version string	So many options...
Component hash	Cryptographic property
Unique identifier	UUID? GUID?
Relationship	"Included in" Others? "Derived from?"

Example 1

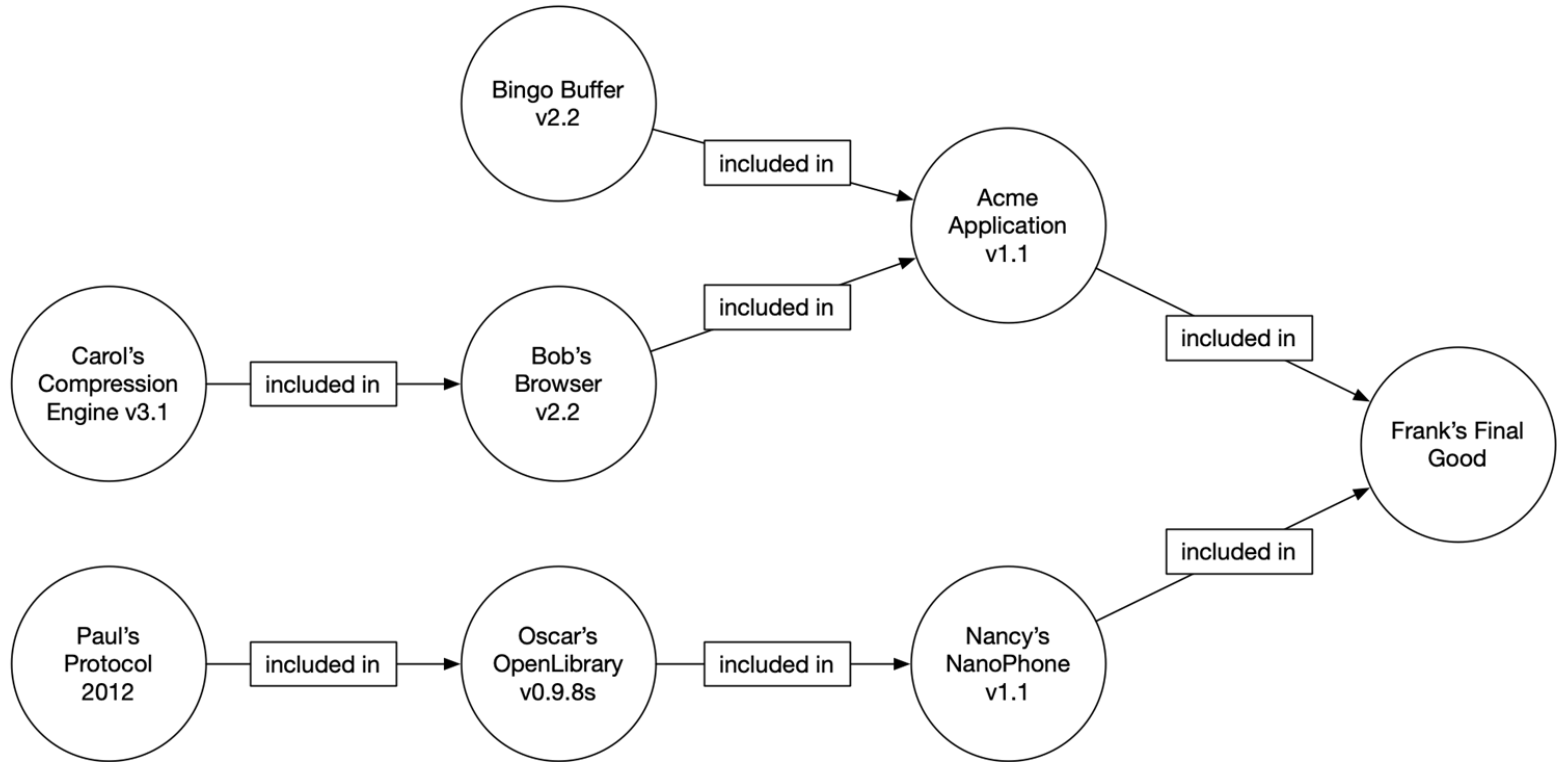
Possibly a directed
acyclic graph

Also works as a
table

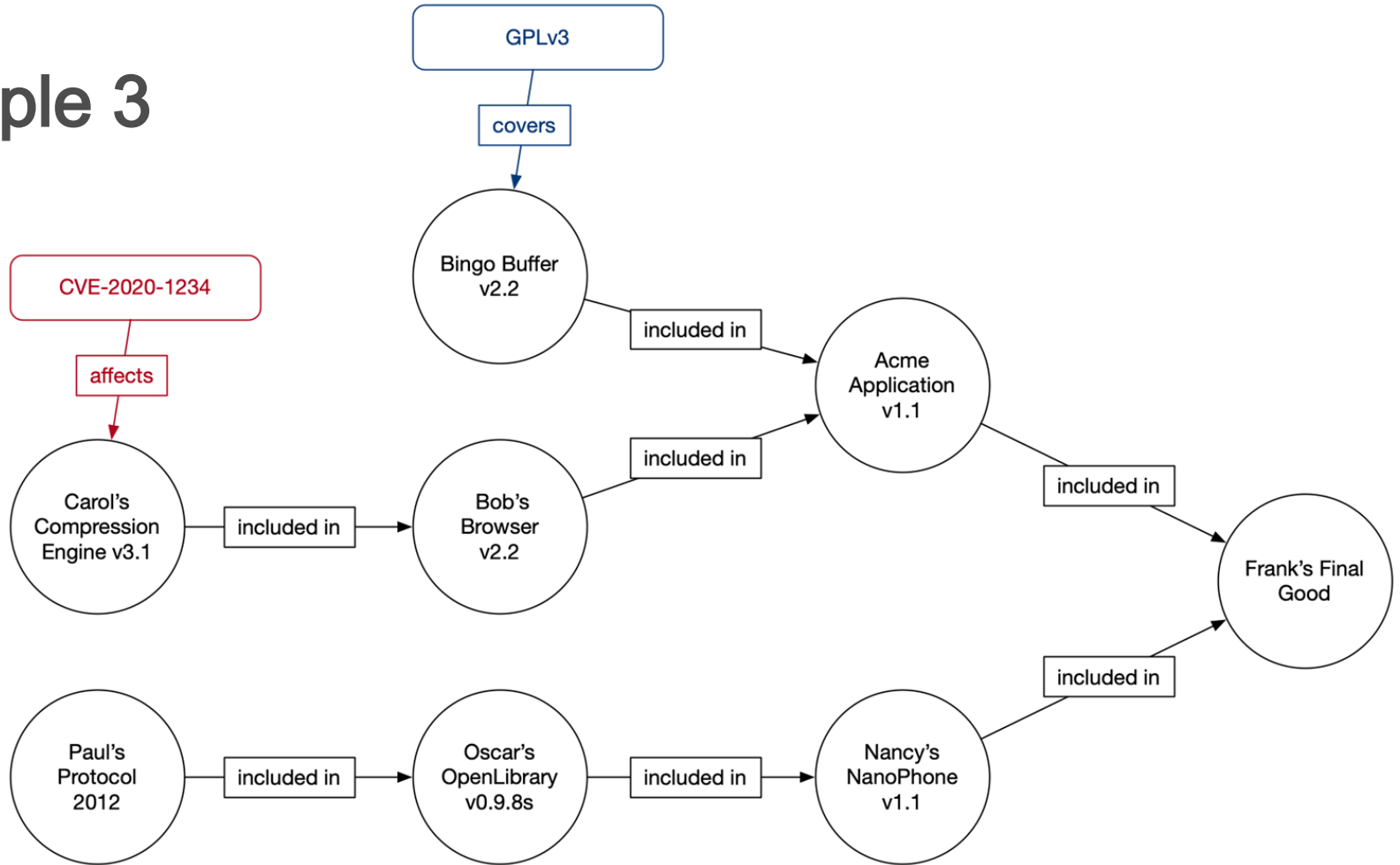


Component Name	Supplier Name	Version String	Author	Hash	UID	Relationship
Application	Acme	1.1	Acme	0x123	234	Self
--- Browser	Bob	2.1	Bob	0x223	334	Included in
--- Compression Engine	Carol	3.1	Acme	0x323	434	Included in
--- Buffer	Bingo	2.2	Acme	0x423	534	Included in

Example 2



Example 3





SBOM Processes

Ask for SBOM from suppliers

Create and provide SBOM to consumers (users, customers)

While not ideal, the SBOM author does not have to be the component supplier

- Partial SBOM is better than no SBOM

Use existing development and package management systems where available

Options under active development (and use) for:

- Component identification
- SBOM format
- Advertisement, discovery, and exchange

SBOM Processes

Changes to components drive changes to SBOMs

- New products, updates, upgrades, patches

Model supports many levels of component abstraction (e.g., operating system, installer, package, files)

- Target is binary, compiled, packaged software, a unit that is transferred from supplier to consumer
- Hardware and source code are not excluded, but not the primary focus

SBOM Challenges

Globally identifying software components, suppliers

- With adequate uniqueness

Sharing and exchange

- Across different classes of device, types of software, sectors

Conveying vulnerability status

- Ripple20, URGENT/11

Incomplete, non-authoritative SBOM information

Vulnerability Status

SBOM alone isn't
enough: Enter "VEX"

SBOMs are useful

- Give lots of key information about what is inside a particular device so you can quickly identify if a product may be affected

But... need another key to the puzzle

- You don't know how exploitable a particular vulnerability may be in a specific device, in a particular environment, until the supplier evaluates the "mechanism of action" and determines if it is applicable
- Often also depends how (and how much of) the affected software component is used in a given device

Example: New ransomware has been identified!

- A new ransomware has been identified! Alerts posted on information sharing channels
- How does it work?
 - Malware has capability to scan port TCP 445 (Server Message Block/SMB) and exploit a new vulnerability discovered in Windows 10
- User wants to know:
 - Am I affected and where?
 - Where do I focus my energies?



Let's look at how SBOM and "VEX" differ...

"I have an SBOM, so I can..."

- Know what components are in device so I know this device uses Windows 10
- Know that the version of Win10 used is vulnerable

But...

- Don't know if port is open
- Don't know if firewalls on device can block malware

SBOM



"I have a VEX document, so I can..."

- Know if the product is actually vulnerable
- More details related to approached I can take to reduce risk while I wait for a patch. (e.g., maintain firewall)

"VEX"



End

In Conclusion

- SBOMs are useful tools for managing security risks of systems
- But managing and using SBOMs effectively and efficiently can be complicated
- NTIA is working on solutions to address broad SBOM challenges
 - This includes sectorspecific efforts