APH10

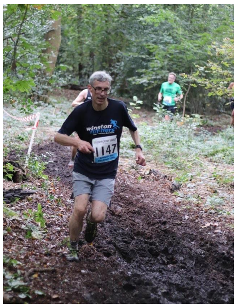
Taming the Supply Chain

March 2024

Anthony Harrison (Founder)
anthony@aph10.com

APH10

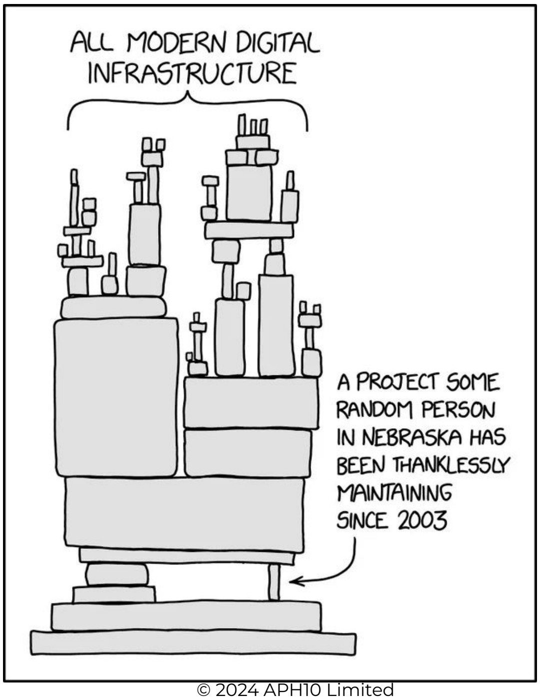
- A career delivering mission critical solutions across multiple sectors
- Founder and Director APH10
- Open Source Software
- STEM Ambassador
- Mentor



CoderDojo

How can the supply chain be disrupted?

Traditional Supply Chain	Software Supply Chain


ALL MODERN DIGITAL INFRASTRUCTURE
A PROJECT SOME RANDOM PERSON IN NEBRASKA HAS BEEN THANKLESSLY MAINTAINING SINCE 2003

https://xkcd.com/2347/

Workshop setup

git clone https://github.com/anthonyharrison/bsideslancs2024.git

cd bsideslancs2024

sh install.sh

sbom4files

```
sbom4files [-h] [-d DIRECTORY] [-p PROJECT] [-r] [-i IGNORE] [--debug]
[--sbom {spdx,cyclonedx}] [--format {tag,json,yaml}] [-o OUTPUT_FILE] [-V]

Input:

-d DIRECTORY, --directory DIRECTORY              Directory to be scanned

-p PROJECT, --project PROJECT                    Name of project

-r, --recurse                                    Recurse directories

-i IGNORE, --ignore IGNORE      Comma separated list of extensions to ignore
```

sbom4files

```
sbom4files [-h] [-d DIRECTORY] [-p PROJECT] [-r] [-i IGNORE] [--debug]
[--sbom {spdx,cyclonedx}] [--format {tag,json,yaml}] [-o OUTPUT_FILE] [-V]

Output:

--sbom {spdx,cyclonedx} specify type of sbom to generate (default: spdx)

--format {tag,json,yaml} format for SPDX software bill of materials (sbom)
(default: tag)

-o OUTPUT_FILE, --output-file OUTPUT_FILE output filename (default: output
to stdout)
```

APH10

sbom4files

```
sbom4files –directory example1 –project “BSides Test1”
```

APH10

sbom4files

```
sbom4files -directory example1 -project “BSides Test1” -sbom
cyclonedx -format json
```

sbom4files

```
sbom4files -directory example1 -project “BSides Test1” -sbom
cyclonedx -format json -recurse
```

sbom4python

```
sbom4python [-h] [-m MODULE] [--exclude-license] [--include-file] [-d]
[--sbom {spdx,cyclonedx}] [--format {tag,json,yaml}] [-o OUTPUT_FILE] [-g
GRAPH] [-V]

Input:

-m MODULE, --module MODULE    identity of python module

--exclude-license            suppress detecting the license of components

--include-file               include reporting files associated with module
```

sbom4python

```
sbom4python [-h] [-m MODULE] [--exclude-license] [--include-file] [-d]
[--sbom {spdx,cyclonedx}] [--format {tag,json,yaml}] [-o OUTPUT_FILE] [-g
GRAPH] [-V]

Output:

--sbom {spdx,cyclonedx} specify type of sbom to generate (default: spdx)

--format {tag,json,yaml} format for SPDX software bill of materials (sbom)
(default: tag)

-o OUTPUT_FILE, --output-file OUTPUT_FILE output filename (default: output
to stdout)

 -g GRAPH, --graph GRAPH     filename for dependency graph
```

sbom4python

```
sbom4python -module sbom4python
```

What can go wrong with the software supply chain?

sbomaudit

```
usage: sbomaudit [-h] [-i INPUT_FILE] [--offline] [--cpecheck] [--purlcheck]
[--disable-license-check] [--age AGE] [--maxage MAXAGE] [--allow ALLOW]
[--deny DENY] [--verbose] [--debug] [-o OUTPUT_FILE] [-V]

Input:

  -i INPUT_FILE, --input-file INPUT_FILE            Name of SBOM file

  --cpecheck            check for CPE specification

  --purlcheck           check for PURL specification

  --disable-license-check disable check for SPDX License identifier
```

sbomaudit

```
usage: sbomaudit [-h] [-i INPUT_FILE] [--offline] [--cpecheck] [--purlcheck]
[--disable-license-check] [--age AGE] [--maxage MAXAGE] [--allow ALLOW]
[--deny DENY] [--verbose] [--debug] [-o OUTPUT_FILE] [-V]

Input:

  --age AGE             minimum age of package (as integer representing
days) to report (default: 0)

  --maxage MAXAGE       maximum age of package (as integer representing
years) to report (default: 2)

  --allow ALLOW         Name of allow list file

  --deny DENY           Name of deny list file
```

sbomaudit

```
sbomaudit  -i example2/sboms/sbom1.spdx
```

Dependency Pining

Strategy	Pros	Cons
Fixed versions		
Dynamic version		

sbomaudit

```
sbomaudit  -i example2/sboms/sbom1.spdx -deny
example2/policy/project.conf
```

sbomdiff

```
usage: sbomdiff [-h] [--sbom {auto,spdx,cyclonedx}] [--exclude-license] [-d]
[-o OUTPUT_FILE] [-f {text,json,yaml}] [-V] FILE1 FILE2

SBOMDiff compares two Software Bill of Materials and reports the
differences.

positional arguments:

  FILE1                 first SBOM file

  FILE2                 second SBOM file
```

sbomdiff

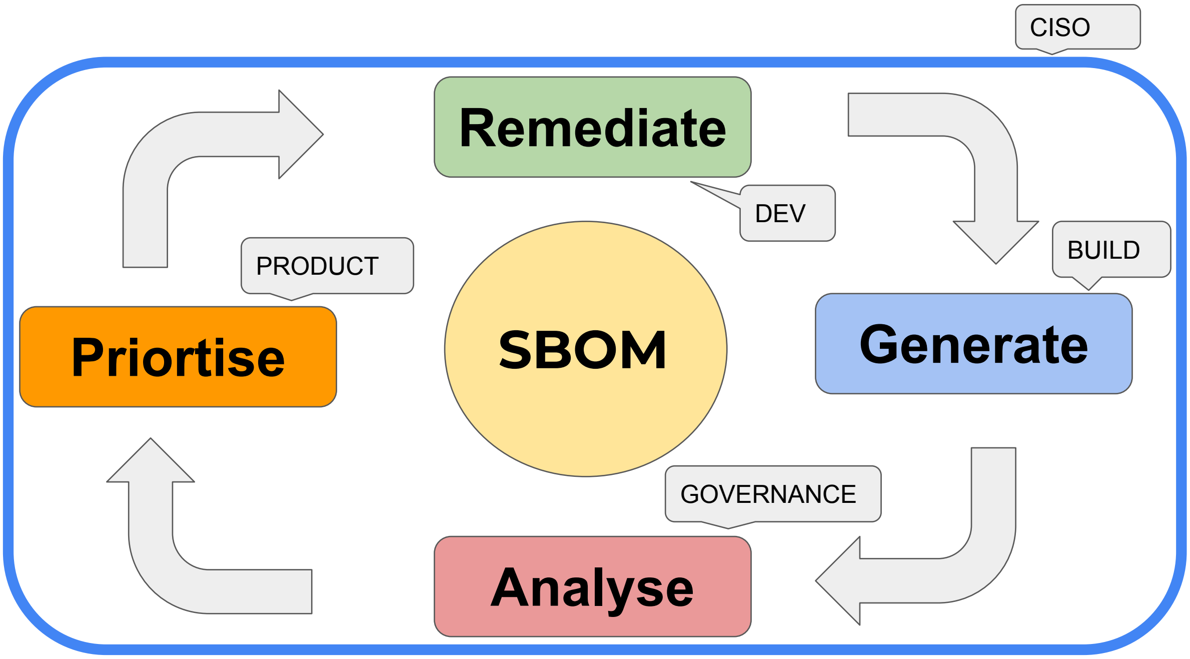
```
sbomdiff  example2/sboms/sbom1.spdx
example2/sboms/sbom2.json
```

sbom2doc

```
usage: sbom2doc [-h] [-i INPUT_FILE] [--debug] [--include-license] [-f
{console,json,markdown,pdf}] [-o OUTPUT_FILE] [-V]

Input:

  -i INPUT_FILE, --input-file INPUT_FILE            Name of SBOM file

Output:

  --debug               add debug information

  --include-license     add license text

  -f {console,json,markdown,pdf}, --format {console,json,markdown,pdf}

  -o OUTPUT_FILE, --output-file OUTPUT_FILE
```

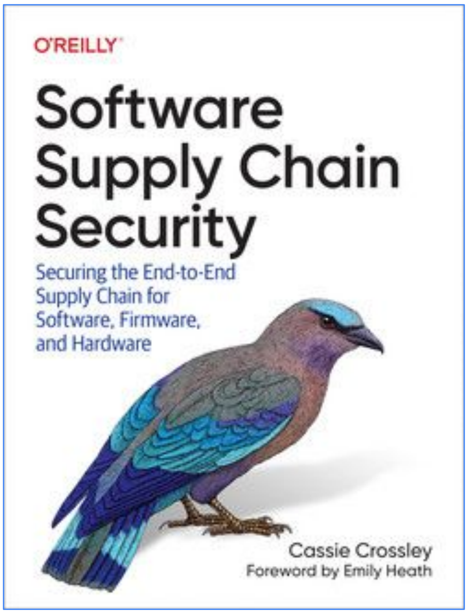
sbom2doc

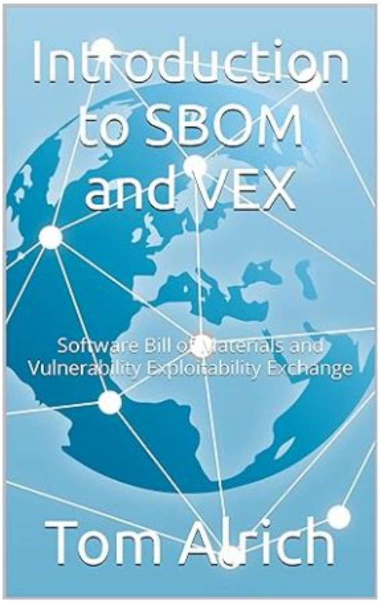
```
sbom2doc -i example2/sboms/sbom1.spdx
```

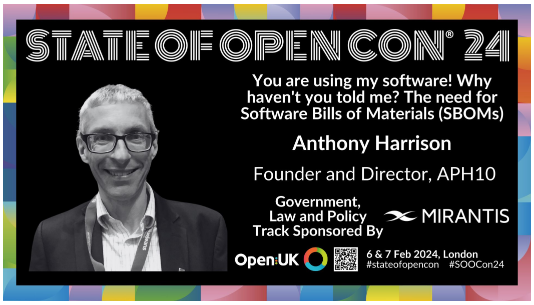


SBOM Types

- Design
  - Represents the software to be produced
- Source
  - Created from source repository. Often produced as part of SCA tool chain
- Build
  - Represents a releasable product resulting from a build process (e.g. an executable)
- Analysed
  - Represents a set of products e.g., executables, packages, containers, and virtual machine images after a build
- Deployed
  - Represents an inventory of all artefacts installed onto a system
- Runtime
  - Identifies the components executing within a system


O'REILLY
Software Supply Chain Security
Securing the End-to-End Supply Chain for Software, Firmware, and Hardware
Cassie Crossley
Foreword by Emily Heath


Introduction to SBOM and VEX
Software Bill of Materials and Vulnerability Exploitability Exchange
Tom Alrich



STATE OF OPEN CON® 24

You are using my software! Why haven't you told me? The need for Software Bills of Materials (SBOMs)

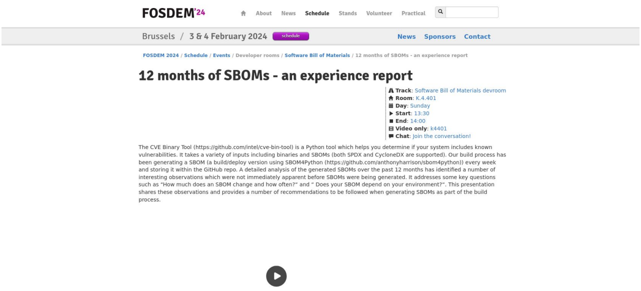
Anthony Harrison

Founder and Director, APH10

Government, Law and Policy Track Sponsored By MIRANTIS

OpenUK

6 & 7 Feb 2024, London #stateofopencon #SOOCon24



FOSDEM'24
About
News
Schedule
Stands
Volunteer
Practical
Brussels / 3 & 4 February 2024
schedule
News
Sponsors
Contact
FOSDEM 2024 / Schedule / Events / Developer rooms / Software Bill of Materials / 12 months of SBOMs - an experience report

12 months of SBOMs - an experience report

Track: Software Bill of Materials devroom
Room: K.4.401
Day: Sunday
Start: 13:30
End: 14:00
Video only: k4401
Chat: Join the conversation!

The CVE Binary Tool (https://github.com/intel/cve-bin-tool) is a Python tool which helps you determine if your system includes known vulnerabilities. It takes a variety of inputs including binaries and SBOMs (both SPDX and CycloneDX are supported). Our build process has been generating a SBOM (a build/deploy version using SBOM4Python (https://github.com/anthonyharrison/sbom4python)) every week and storing it within the GitHub repo. A detailed analysis of the generated SBOMs over the past 12 months has identified a number of interesting observations which were not immediately apparent before SBOMs were being generated. It addresses some key questions such as "How much does an SBOM change and how often?" and " Does your SBOM depend on your environment?". This presentation shares these observations and provides a number of recommendations to be followed when generating SBOMs as part of the build process.

https://www.youtube.com/watch?v=WrUVKqKaq1Y

https://fosdem.org/2024/schedule/event/fosdem-2024-1896-12-months-of-sboms-an-experience-report/

Thank you

Anthony Harrison

anthony@aph10.com

APH10