

EU CRA: Time to Engineer Up!

Up Your Engineering Game:
Why You Need SBOMs and Dependency Management

Peter N. M. Hansteen
19 February 2026

```
{
  "schema": "http://cyclonedx.org/schema/bom-1.6",
  "bomFormat": "CycloneDX",
  "specVersion": "1.6",
  "serialNumber": "urn:uuid:d74b876a-1a1b-4b9b-83",
  "version": 1,
  "metadata": {
    "timestamp": "2025-03-10T09:55:03+01:00",
    "tools": {
      "components": [
        {
          "type": "application",
          "author": "anchore",
          "name": "syft",
          "version": "1.20.0"
        }
      ]
    }
  },
  "component": {
    "bom-ref": "af63bd4c86077f1",
    "name": "syft"
  },
  "components": [
    {

```

tiesto

<https://nxdomain.no/sbom.pdf>



EU CRA: It's Later Than You Think, Time to Engineer Up!

- *December 12 2027, it's too late.*
- On December 11 2027, the European Union Cyber Resilience Act (CRA) enters fully into force

=> Products with digital elements must come with **full overview of** and **insight into**

- Description and origin of all **components**
- Description and origin of all **dependencies**

Or, no CE mark of approval for you or your product

Reporting required for all vulnerabilities and incidents will have been required since *September 11, 2026*.

[Full text article: *EU CRA: It's Later Than You Think, Time to Engineer Up!*]

Upping Your Engineering Game

- Are you
a coder, or
an engineer who codes?
- This is about upping your engineering game
 - Taking your software engineering to “real engineering” level
 - Focus on doing things right, proper practices for quality engineering

The anecdote about the iron ring worn by Canadian engineers is useful to calibrate your mindset and level of ambition.

Dear Developer, do you know what your code does?

You wrote it, but do you know **all** it does? –

TL; DR:

Your code has dependencies.

The rest of the world, including customers recently started to care about which ones.



But first, a bit of history.

“Just a bit of typing”

Software is a new phenomenon.

Poorly understood by the public

Considered “just a bit of typing”

Not important in itself, but a necessary evil

Poorly understood by non-techies

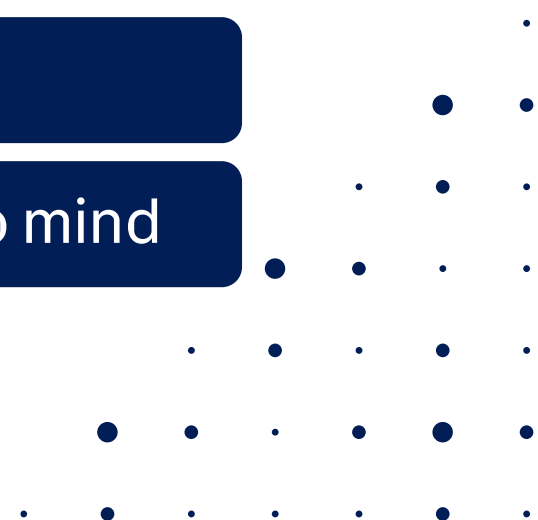
But then suddenly software turned important

First, the Internet happened and raised our profile

Bugs in local code started biting

Bugs in dependencies started biting – log4j comes to mind

Process bugs started happening – Solarwinds SUNBURST comes to mind



Dependencies became a thing

- Next up, memes like ->
(see [XKCD #2347](#), but also the [explainer](#))

=>

Supply chain management

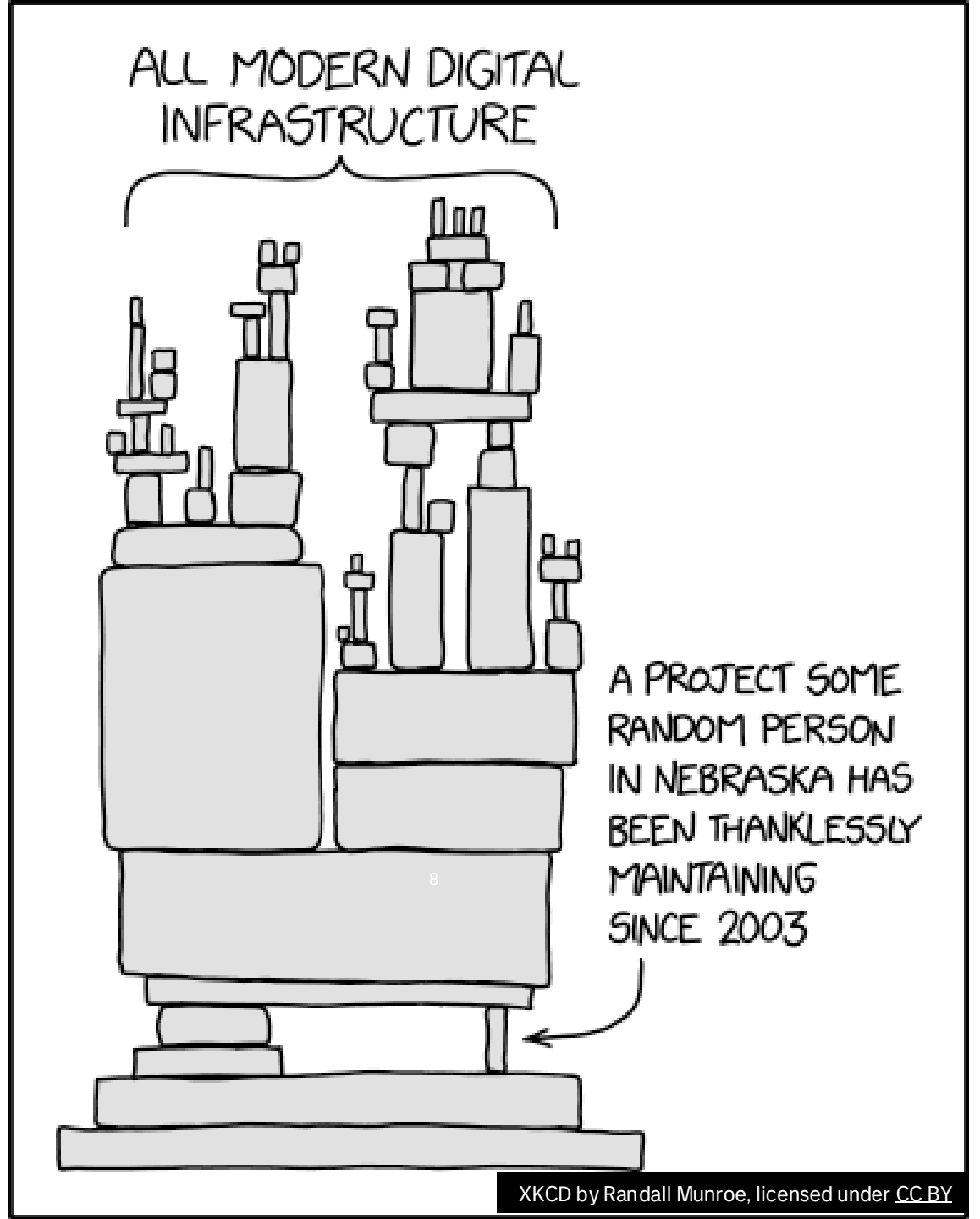
and

dependency management

Internal

came into focus

!! Known your components and dependencies !!



Dependencies became a thing

- Next up, memes like -> (modernized, based on [XKCD #2347](#), but also the [explainer](#))

=>

Supply chain management

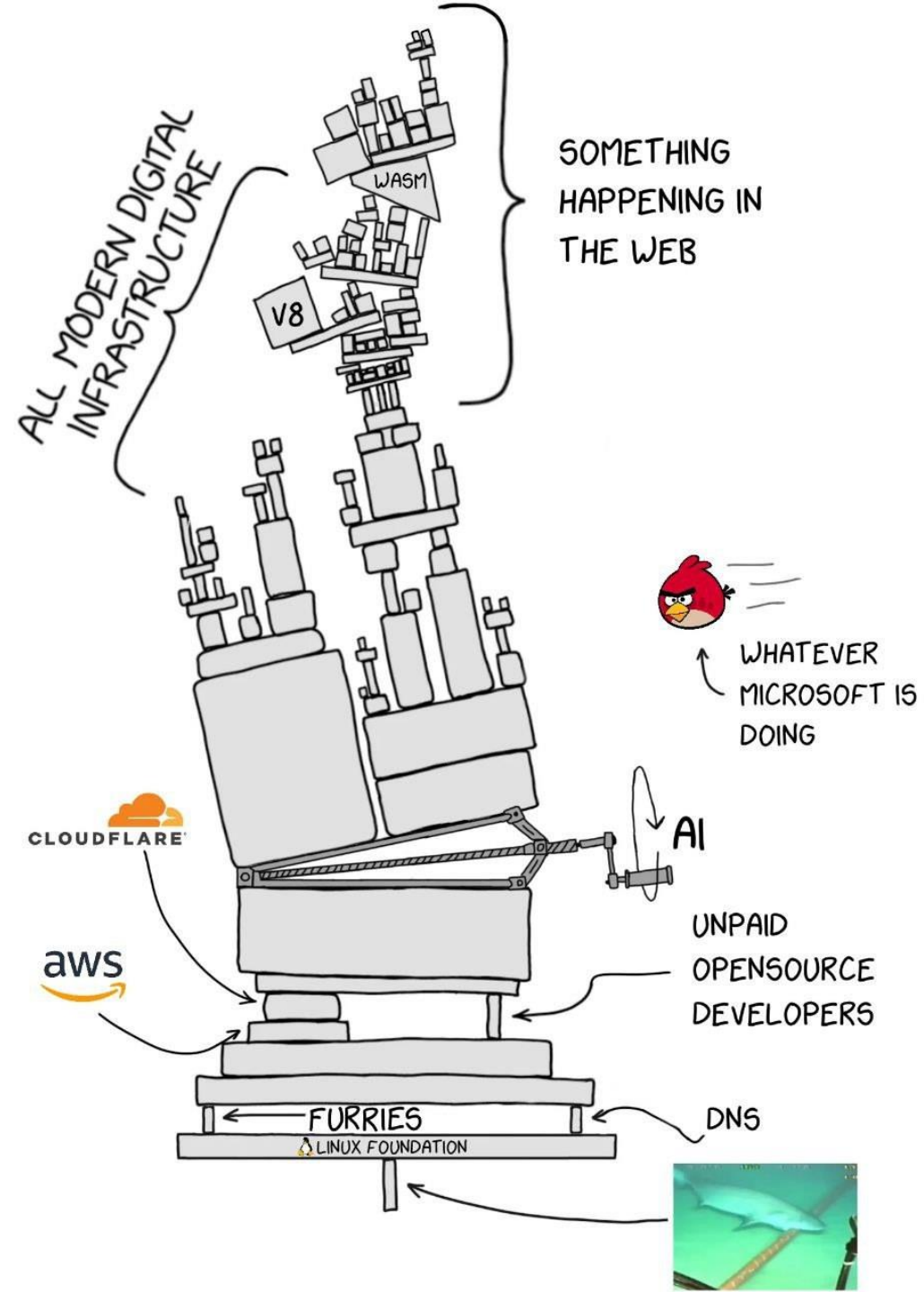
and

dependency management

came into focus

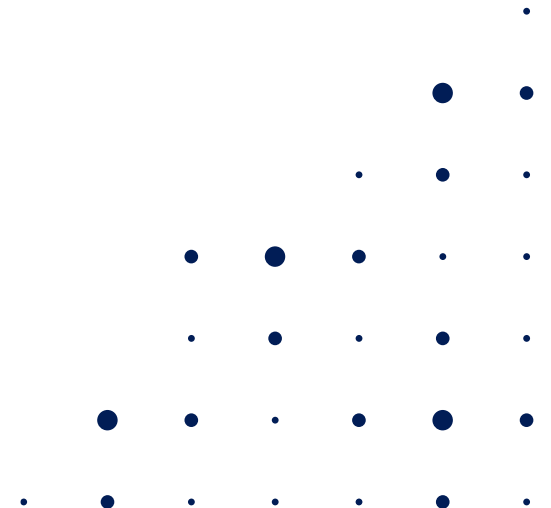
Internal

!! Known your components and dependencies !!



No project is an island

- We write software
- Which depends on other software
- Which interacts with other software
- Which again interacts with other components (hardware, humans)
- To run important stuff
- Nothing exists in actual isolation – No project is an island



Learn from those who build important things

- What to those who build important things do?
- Real engineers with real plans are required to submit a “Bill of Materials” aka BOM for each delivery

Internal

HMS QUEEN ELIZABETH



- 1 Aviation store
- 2 F35 Lightning II
- 3 Phalanx automated close-in weapons system
- 4 Forward island bridge
- 5 Hangar deck
- 6 Long range radar
- 7 Forward engine and gas turbine uptakes
- 8 Forward aircraft lift
- 9 Stern helicopter
- 10 After island emergency conning bridge
- 11 Mainmast
- 12 Medium range radar
- 13 Communication ducts
- 14 After engine room and gas turbine uptakes
- 15 Living control position
- 16 After aircraft lift
- 17 Chinook helicopter
- 18 Automatic small calibre gun
- 19 Inflatable life-raft storage
- 20 Forward mooring deck
- 21 Junior rates' sea berth cabins
- 22 Junior rates' showers and toilets
- 23 Ship's office complex
- 24 Pyrotechnic compartment
- 25 Forward gas turbine space
- 26 Forward engine down-takes
- 27 Fire protection system
- 28 Hangar forward bay
- 29 Forward hangar doors
- 30 Air evacuation complex
- 31 Mass evacuation system
- 32 HR police office and canteen
- 33 After engine down-takes
- 34 After gas turbine space
- 35 After hangar doors
- 36 Air filtration units
- 37 Hangar aft bay
- 38 Hangar aft bay
- 39 Aft mooring deck
- 40 Starboard mooring deck
- 41 Port mooring deck
- 42 Anchor port and starboard
- 43 Water ballast compartment
- 44 Chem locker tank
- 45 Gym
- 46 Junior rates' recreation space
- 47 Mission systems office
- 48 Mission systems complex
- 49 Forward engine room uptakes
- 50 Battery
- 51 Pipe passage
- 52 Junior rates' galley
- 53 Junior rates' dining hall
- 54 MARPAC common spaces
- 55 Low voltage distribution compartment
- 56 After engine room uptakes
- 57 Hospital area
- 58 Ward area
- 59 General medical area
- 60 AGN and ship control centre
- 61 Senior rates' dining hall
- 62 Officers' and senior rates' galley
- 63 Wardroom
- 64 Wardroom annex
- 65 Head of department cabins
- 66 Flag officer and commanding officers' galley
- 67 Flag officer and commanding officers' dining
- 68 Commanding officer's suite
- 69 Flag officer's suite
- 70 Junior rates' port and starboard
- 71 Busbar box
- 72 Auxiliary machinery space
- 73 Naval stores complex
- 74 Aerial tank
- 75 Forward engine room
- 76 Fresh water tanks
- 77 Stateroom compartment (port and starboard)
- 78 Stateroom (port and starboard)
- 79 Heat exchanger tank (port and starboard)
- 80 Bilge keel
- 81 Void
- 82 After engine room
- 83 Officers' baggage store
- 84 Propeller (port and starboard)

aircraft carrier
ALLIANCE
Delivering the Nation's Flagships

HMS QUEEN ELIZABETH

inside HMS QUEEN ELIZABETH



© Crown Copyright 2010. Aircraft Carrier Alliance (RMA) Ltd Box 85, Llandudno, Gwynedd, LL50 2YU. All rights reserved. This document is the property of the Crown and is not to be reproduced in any form without the written permission of the Crown. The information contained herein is for the use of the public only and is not to be used for any other purpose. The information contained herein is not to be used for any other purpose. The information contained herein is not to be used for any other purpose.

What do engineers do?

- Create a plan
- A Bill of Materials (BOM) is required for all deliveries
- The BOM lists all component parts
- The BOM typically also references and serves as reference for maintenance docs

Internal

OM and can make changes.

Upgrade to Team

Assembly Name: PUMP S55-Main-Boat Pump Assembly

Revisions (Latest state)

Production planning:

Order BOMs

Part and Catalogs

↑ ↓	☰	☐ Text wrap	↶ ↷	Where used	Composed of	☐ Display Totals	↶ Property order	☰ Filter	ⓘ Info panel	
..	..	Material	Quantity	Part ...	Link to...	Description	Mass (..	Den..	Supplier V..	Unit Cost
	Part	Stainless Steel	1	Pump 1_2 ...	Boat Pump...	Pump body	3.56	0.28	☐	40
	Part		1	D-85-STV...	Boat Pump...	Pump Cap ZZZZ	0		☒	8.5
	Part	Stainless Steel	5	SH100	Socket butt...	Socket Head S...	0.01	0.28	☒	1.2
	Part	Stainless St...	1	D85-55XU...	Boat Pump...	Gasket/Spacer ...	0.01	0.28	☒	3.2
	Part	Stainless St...	1	Imp 1-2-KK	Boat Pump...	Impeller descrip...	0.57	0.28	☐	13.3
	Part	Stainless St...	1	L-104	Boat Pump...	Shaft	0.34	0.28	☐	50.4
	Part	Stainless St...	1	6203-02-0...	Boat Pump...	Impeller part	0.17	0.28	☐	20.3
	Part	Silicone Rub...	1	3222	Boat Pump...	Seal	0.01	0.04	☒	3.2
	Part	Stainless St...	1	6203-01-0...	Boat Pump...	Radial bearing	0.17	0.28	☐	20
	Part	Aluminum - ...	1	HDWRE C...	Boat Pump...	Circlip 1.25	0	0.1	☐	0.4
	Part		4	NM100C2	Boat Pump...	Socket Head S...	0		☐	0.5

Libre software has package management already



In the Open source / Free software / Libre Software world, we have a long history of package management systems that take care of *runtime dependencies* for installation and configuration



Those in turn depend on build systems that take care of *buildtime dependencies* and to generate installable packages



The information to build a Software Bill of Materials is right there in our source code



In addition, we tend to have runtime vulnerability scanners to look for bugs and reported weaknesses and CVEs

Introducing: A Software Bill of Materials (SBOM)

- Nuts and bolts aside, the legal framework is:

US Executive Order 14028 of May 12, 2021, *Improving the Nation's Cybersecurity*, (summarized) – USA: [NTIA.gov](https://www.ntia.gov)

or

EU Cyber Resilience Act (CRA), also more reader friendly at the Cyber Resilience Act start page (EU/EEA)

- This legislation is in force or is soon to be, main concepts are taking the *dependency* information we have and present for compliance in actionable form. Inclusion of security info such as CVEs much appreciated.
- The main takeaway:

The information we need is in our code; we need to make generating for compliance effortless and painless.

We're real engineers now, Sparky! We have tools!

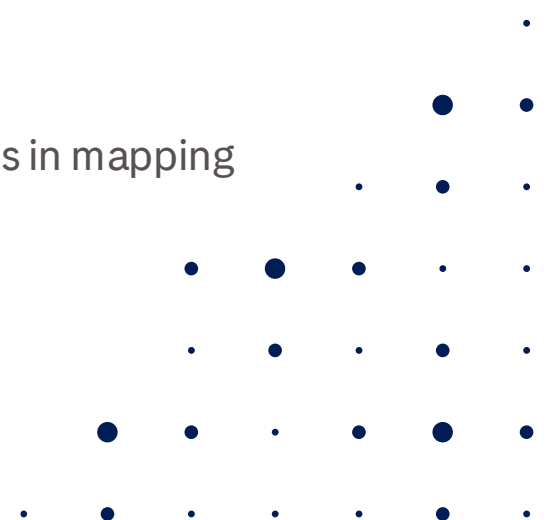
- Mostly from open source circles, two SBOM specs with tool suites emerged:

Open Worldwide Application Security Project (OWASP) CycloneDX

and

Linux Foundation's System Package Data Exchange specification (SPDX)

- For tools and guidance on both, [awesome-sbom](#) (github.com) is a good place to start.
- Conversion between the two formats is common and available from several tools, **but** there are gaps in mapping between the two



Tools and how to use them

- Using syft to generate and bomber to present, go to a project directory and do

```
$ syft . -s all-layers -o cyclonedx-xml | xq
```

or the json equivalent

```
$ syft . -s all-layers -o cyclonedx-json | jq
```

- If you want a html report with known vulnerabilities

```
$ syft . -o cyclonedx-json | bomber scan --provider ossindex --output=html
```

Note: For this particular command to work, you also need to supply provider login credentials (available with free registration), see the [Bomber provider documentation](#).

Your SBOM, the build artifact

You probably want your pipeline to produce SBOM build artifacts.

For inclusion in a CI/CD pipeline you may want

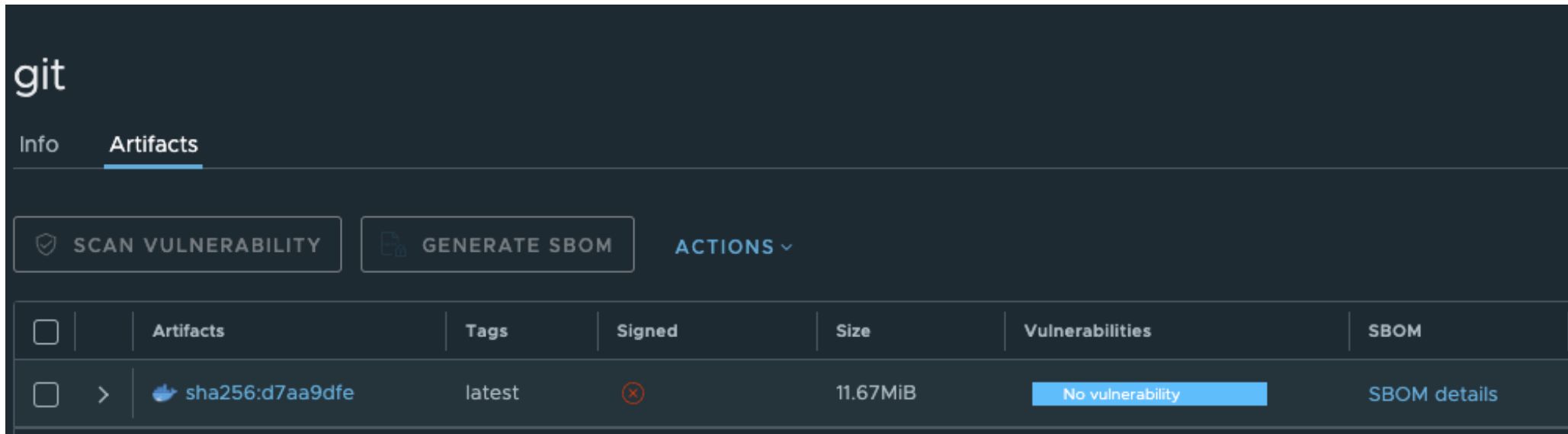
```
# Make sure you include the - character at the end of the command.  
# This triggers bomber to read from STDIN  
syft packages . -o cyclonedx-json | bomber scan --provider ossindex --output json -
```

Note: For this particular command to work, you also need to supply provider login credentials (available with a free registration), see the [Bomber provider documentation](#).

Also, reports and conversions, see the docs for your toolset or get inspired by looking at [awesome-sbom](#)

Your tools may already have (some of) this

- Some tools have SBOM generating and presentation already. Harbor has automatic SBOM generation on image push using trivy:



- On the receiving end, there is *Dependency Tracker* (<https://dependencytrack.org/>) among others

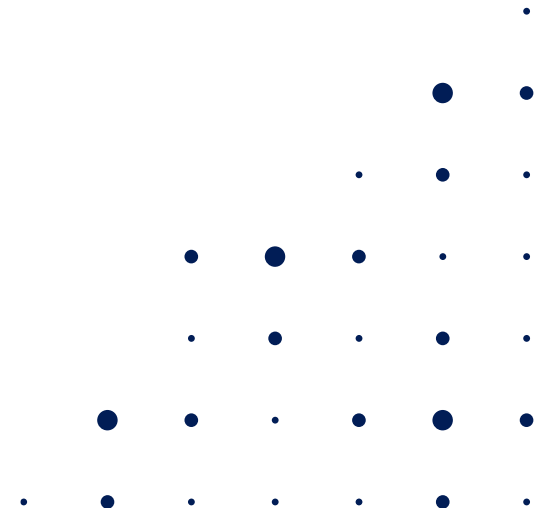
Track your dependencies on the fly

- When you have the tools in hand, when a customer asks about your tooling:

you could do

```
$ cd myproject  
$ cdxgen .
```

and get a `bom.json` with all your customer wants to know (see the article)



There is more -



Your SBOM-savvy stakeholders may already have appropriate processing tools at their end. Poke them!



Other aspects of BOM generation are being worked on, including
OBOM (Operating system Bill of Materials)
SaaSOM (Software as a Service Bill of Materials)
CBOM (Cryptography Bill of Materials)
... and others



There is more to SBOMs out there, to impress colleagues and customers!

Now It's Your Turn: Get The Tools

- For further exploring, you need to get the tools:
- cdxgen lives at <https://github.com/CycloneDX/cdxgen>
- syft can be found at <https://github.com/anchore/syft>
- bomber is at <https://github.com/devops-kung-fu/bomber>

Follow the install instructions for each, On Linux, they are likely in your package system. On macOS, use brew,

```
$ brew install $toolname
```

Again, follow the install instructions for your environment.

Or use

```
$ git clone $toolname
```

(As one does)

Tools in Hand, Dig Into a New Project

Find a project. Do you have one of your own? Get the code.

Go to the project main directory. Run

```
$ cdxgen .
```

Watch the output. It will be useful to run this in a script(1) session.

This is where the fun starts -

Tools in Hand, Dig Into a New Project (2/2)

Watch the output. It will be useful to run this in a script(1) session.

- Find out:

What is the number of dependencies for this code base? How many direct dependencies? How many indirect ones (dependencies of dependencies)?

- Does the code base itself have any known problems, reported as CVEs? How many for the dependencies?

You may choose to inspect the tools themselves.

Remember, there may be challenges, like in this session. Don't let noisy errors discourage you!

This is where you pick up your engineering game

Resources for further reading

- Linux Foundation Training: Automating Supply Chain Security: SBOMs and Signatures (LFEL1007) is a short but information- and reference-filled introduction (free, requires registration, gives you a badge at the end)
Understanding the EU Cyber Resilience Act (CRA) (LFEL1001) Focused on the EU CRA, gives an overview with lots of useful references, nominally a 1 hour course worth taking
- European Union Cyber Resilience Act (CRA) resources page at the Open Source Security Foundation (OpenSSF)
- The Software Bill of Materials home page at NTIA is the mother ship of SBOM documentation
- Browse OWASP CycloneDX for all things about the CycloneDX specification and related tools, also their CycloneDX tool center
- Browse the System Package Data Exchange specification (SPDX) for all things SPDX (supported by the Linux Foundation), including copious linked reference material
- awesome-sbom is a curated list of SBOM tools and resources
- Awesome CRA Compliance is a curated list of CRA compliance resources
- EU residents will want to poke around the Cyber Resilience Act site for reference
- Brewing Transparency: How OWASP's TEA Is Revolutionizing Software Supply Chains is a summary of recent work on OWASP Transparency Exchange API (TEA)
- SBOM buyer's guide: 8 top software bill of materials tools to consider is a readable overview of (some) SBOM tools
- Olle Johansson's FOSDEM presentations are among several good SBOM talks at that conference (search the site for more)
- Peter N. M. Hansteen: Open Source in Enterprise Environments - Where Are We Now and What Is Our Way Forward (2022, also here) has some insights on how open source software plays a crucial role in enterprise environments and elsewhere
- Peter N. M. Hansteen: No Project Is an Island: Why You Need SBOMs and Dependency Management (also here)
- Peter N. M. Hansteen: EU CRA: It's Later Than You Think, Time to Engineer Up! (this presentation in article form, also here, slides)

Thank you

Peter N. M. Hansteen
10 February 2026

