



sbom-cve-check: Lightweight Python tooling for out-of-build CVE analysis of SPDX3 SBOMs

Benjamin Robin

benjamin.robin@bootlin.com

FOSDEM 2026

© Copyright 2004-2026, Bootlin.

Creative Commons BY-SA 3.0 license.

Corrections, suggestions, contributions and translations are welcome!





Benjamin Robin

- ▶ Embedded Linux engineer at **Bootlin**
- ▶ Bootloaders, kernel development, Buildroot and Yocto integration
- ▶ Open-source contributor
- ▶ Living in **Lyon**, France



sbom-cve-check



What is it?

- ▶ A **standalone** python tool, with as few dependencies as we could manage
- ▶ Takes as input:
 - an SBOM (currently SPDX 2.2 and SPDX 3.0 formats are supported)
 - (optional) the JSON VEX manifest generated by Yocto's `vex.bbclass` in upstream oe-core since Yocto 5.0.15 "Scarthgap", also in Walnascar and in Styhead.
 - some enrichment databases
- ▶ Looks up which CVEs impact the SBOM, and therefore potentially the systems
- ▶ Outputs vulnerability assessments in a format that can be fed to downstream tools
- ▶ Overall, close to Yocto's `cve-check.bbclass`

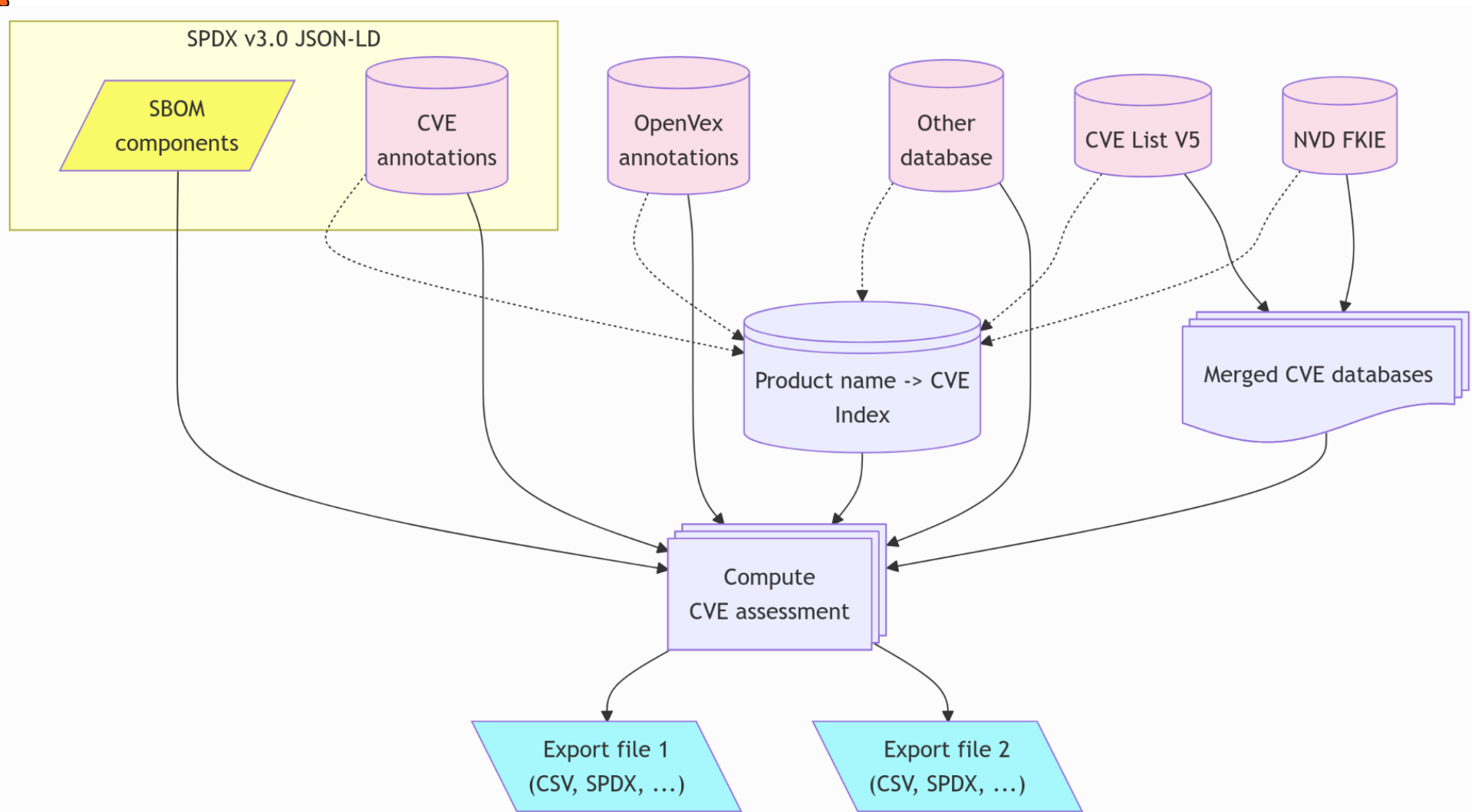


Why is it?

- ▶ Not being aware of which CVEs affect one's stuff is going to get expensive soon with the EU Cyber Resilience Act (CRA).
- ▶ `cve-check` is great, but runs **during a build** of Yocto. This is inconvenient because:
 - CVE checking already takes a while, no need to add build time to it
 - once the software is shipped, building an old version might be unnecessary effort
 - what if we wanted to check artifacts of a build we did not/cannot run?
- ▶ There are very few, if any, analysis tools that support SPDX3.
- ▶ Easy
 - few dependencies
 - easy to install and use
 - no bells, no whistles



Overall design





Input & outputs formats

- ▶ The tool pulls from several CVE databases:
 - **NVD**, from github.com/fkie-cad/nvd-json-data-feeds
 - **CVE List**, from github.com/CVEProject/cvelistV5
- ▶ It supports multiple annotation formats:
 - OpenVEX
 - Yocto's custom format, generated from `vex.bbclass`
 - A simple annotations format stored in YAML files.
 - Annotations provided in SPDX 3.0 SBOM file.
- ▶ It supports the following export formats:
 - SPDX3: Only if the input SBOM is also an SPDX 3.0 file (for now).
 - CSV
 - Yocto's cve-check output format.



Features

- ▶ **sbom-cve-check** can be extended by using custom (external) plugins. This allows to add a new type or format that is not currently supported. For example:
 - A new type of SBOM format
 - A custom CVE annotation format
 - A custom report (export) format (Web page, PDF, ...)
- ▶ Mark a CVE automatically as ignored if affected sources are not compiled.
 - Require to set `SPDX_INCLUDE_COMPILED_SOURCES` to 1 in each recipe that need this extra processing.
 - The CVE database needs to provide the list of affected files: Most of the Linux kernel CVEs have this information.



Sources & documentation

- ▶ The source code is available here: `github.com/bootlin/sbom-cve-check`
 - The source code is under GPLv2
 - Contributions are of course welcome :)
- ▶ The documentation can be consulted from: `sbom-cve-check.readthedocs.io`



Want to try it out?

- ▶ Install the tool:
 - Create a Python virtual environment (recommended)
 - `pip install sbom-cve-check[extra]`
- ▶ Generate artifacts using Yocto (or use existing ones):
 - SPDX v3.0 is generated by default since Yocto Walnascar (5.2).
 - Add `INHERIT += "vex"` in `local.conf`.
- ▶ Retrieve these artifacts from Yocto `deploy` directory:
 - `${IMAGE_NAME}.rootfs.spdx.json`: The SPDX v3.0 SBOM file.
 - `${IMAGE_NAME}.rootfs.json`: File generated by the `vex.bbclass`.
- ▶ Execute:

```
sbom-cve-check --sbom-path ${IMAGE_NAME}.rootfs.spdx.json \  
  --yocto-vex-manifest ${IMAGE_NAME}.rootfs.json \  
  --export-type yocto-cve-check-manifest --export-path out.json
```

Thank you!

Benjamin Robin

benjamin.robin@bootlin.com

Slides under CC-BY-SA 3.0

<https://bootlin.com/pub/conferences/>