Automating license scanning and reporting

Photo by Pankaj Patel on Unsplash

Hello, I’m Thomas Steenbergen 👋

Head of Open Source Program Office (OSPO) @ EPAM
Help orgs with open source management and supply chain security

github.com/tsteenbe
@tsteenbe
linkedin.com/in/tsteenbe

Maintainer of the ORT Project

OSS Review Toolkit
oss-review-toolkit.org

Lead for the SPDX SBOM Security Profile

SPDX
spdx.dev

TODO Steering committee member
Co-founder of TODO Europe

TODO
todogroup.org

Creator/organizer OSPOlogy.live workshops

OSPOLogy

Co-founder of OpenChain Automation WG

OPENCHAIN
openchainproject.org

Other communities I am involved in

OpenSSF
OPEN SOURCE SECURITY FOUNDATION
bitkom

We only use Open Source so we don’t have to care about open source licenses …

- Nameless FOSS user

We forbid all use of copyleft open source so you should not find any issues …

- Nameless FOSS Compliance manager

Why: Open Source Compliance Program

OPENCHAIN
ISO 5230

- **Know your obligations.** You should have a process for identifying and tracking Open Source components that are present in your software

- **Satisfy license obligations.** Your process should be capable of handling Open Source license obligations that arise from your organization’s business practices

Benefits of a robust Open Source Compliance program include:

- Increased understanding of the benefits of Open Source and how it impacts your organization
- Increased understanding of the costs and risks associated with using Open Source
- Increased knowledge of available Open Source solutions
- Reduction and management of infringement risk, increased respect of Open Source developers/owners’ licensing choices
- Fostering relationships with the Open Source community and Open Source organizations

Source: OpenChain training slides: https://github.com/OpenChain-Project/curriculum

What information do you need to gather?

When analyzing Open Source usage, collect information about the identity of the Open Source component, its origin, and how the Open Source component will be used. This may include:

- Package name
- Status of the community around the package (activity, diverse membership, responsiveness)
- Version
- Download or source code URL
- Copyright owner
- License and License URL
- Attribution and other notices and URLs
- Description of modifications intended to be made
- List of dependencies
- Intended use in your product
- First product release that will include the package
- Location where the source code will be maintained
- Possible previous approvals in another context
- If from an external vendor:
- Development team's point of contact
- Copyright notices, attribution, source code for vendor modifications if needed to satisfy license obligations

Source: OpenChain training slides: https://github.com/OpenChain-Project/curriculum

What should we buy? 🤔

Just buy what Amazon, Google, Microsoft use to scan for open source licenses in their code …

- Nameless Executive

Our tool can fully automate your license compliance …

- Sales Rep, SCA Vendor

Questions to ask when comparing SCA tools

- **Which SW components are included?** Most build tools are meant to build code and not to produce an SBOM. As a result, software composition analysis tools on the market generally do a best effort approach thus their SBOMs may differ.

- **Can the tool provide proof for a finding?** A lot of SCA tools show for example OSS licenses in the SBOM but won't show how they came to that license e.g., show the exact files and lines in the source code.

- **Can it do my policy or risk decisions?** Default is to have 1) allow/deny list for licenses and 2) 'ignore' buttons for vulnerabilities but satisfy obligations is not black & white and ignoring issues does not solve them.

- **Can we run in CI with acceptable compliance levels?** Run the tool cost effectively at scale and speed whilst maintaining the right levels of compliance and not overload users with false-positives.

- **Can we edit SBOM?** No tool is 100% correct as real world is ugly, being able to manually fix things is a must-have.

See also: https://linuxfoundation.org/resources/publications/an-open-guide-to-evaluating-software-composition-analysis-tools/

Or in short…

Does the tool enable
data-driven risk-driven
open source governance automation?

An introduction to…
ORT

A FOSS policy automation and orchestration tool
to manage open source in a strategic, safe and efficient manner

github.com/tsteenbe

Solving OSPO challenges via industry collaboration

Collaborate to achieve better, cheaper and faster solutions and move the community forward

Community

Collaborations to run successful/effective open source projects and programs.

TODO

FINOS

Good Governance
an OW2 initiative

CHAOSS

OSPO++

Standards

ISO standards for compliance processes and exchange of FOSS metadata.

OPENCHAIN

SPDX

Processes

Organizations collaborating on designing reference compliance processes based using open source tools.

InnerSource

OpenChain Automation Work Group

Security

Collaborative effort to improve open source software security.

OpenSSF
OPEN SOURCE SECURITY FOUNDATION

Automation

Automate FOSS policy & processes within CI/CD.

OSS Review Toolkit

Tooling Challenges

Our experience when trying to introduce automation:

Missing / incorrect metadata

Source location may not be defined or found. Declared ≠ detected license

No sources available

Simply missing in central repositories

MISSING DATA

Ways of working issues

Devs do not always follow best engineering practices
resulting host issues when trying to automate

Build/dependency tools issues

Not designed to support FOSS reviews
e.g. lacking methods or return inaccurate data

Different build/dependency tools

~30 common build/dependency tools

Large volume of scan results

No tooling is available to automate reviewing large amounts of scan results,
conclude obligations and determine any issues to be resolved within limited timeframe

MISSING TOOLING

Automate your FOSS processes with ORT

- Created with and by the OSS/OSPO community for the community

  Vibrant community of users (mostly automotive OSPOs) using and contributing. ORT maintainers participate in SPDX, OpenChain, OpenSSF and the TODO group.

- Battle-tested solution

  ORT is used in production within multiple large organizations and hundreds of thousands ORT scans have been over the last 6 years

- Tooling + Data + Policy

  Everything you need is included such as large set of open source repository and licensing fixes contributed by various OSPOs

- Supports all the major package managers

  No build tool plugins required, detects out-of-the-box the dependencies of a project.

- Dev Ops integration

  Designed from the beginning for a CI/CD world - integrations available for GitHub, GitLab, Jenkins, Tekton and Bitbucket coming soon.

- Highly customized pipelines

  ORT is implemented as set of libraries (for programmatic use) and exposed via a command line interface (for scripted use)

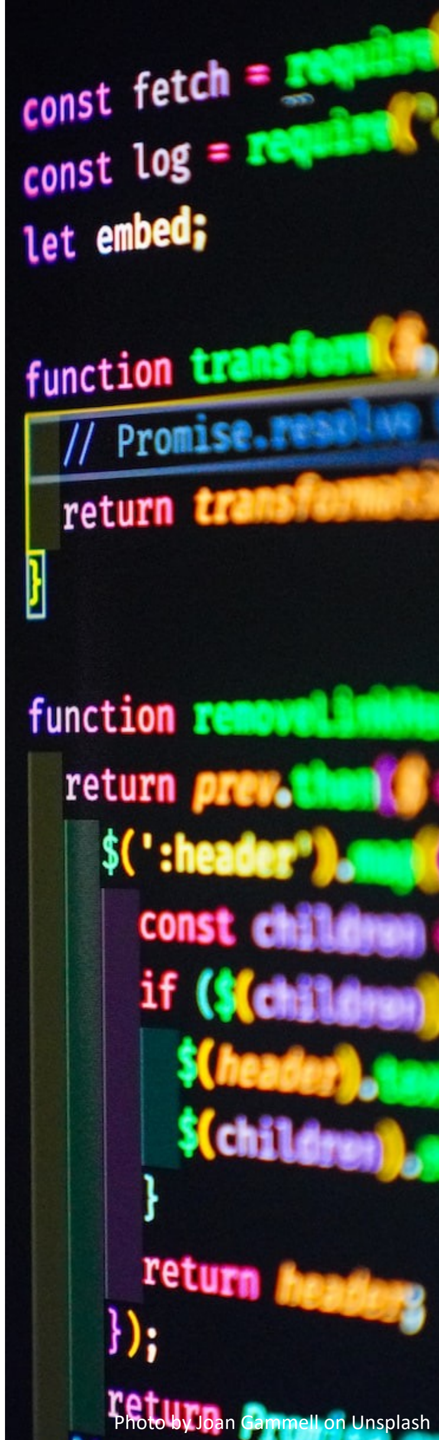


Photo by Joan Gammell on Unsplash

ORT Key Features

- License scanning

  Identifies copyrights and licenses by wrapping existing license / copyright scanners like ScanCode to detect findings in local source code directories.

- Security scanning

  Integrations with OSS security vulnerabilities data feeds from various vendors (Nexus IQ , VulnerableCode, OSS Index and OSV supported).

- Best practices / company standards / InnerSource scanning

  Align software projects across the organization.

- Source code scanning (work in progress)

  Working on partnerships with vendors (FossID, SCANOSS e.a.) to develop integrations to identify published origin of source code and other files

- Policy as Code

  Write your own policy rules on all information collected by ORT whether licensing, security or engineering standards checks

- Generate software bill of materials Notices

  Generate CycloneDX, SPDX 2.2 files, plain text open source notices or your custom result files (via Apache FreeMarker template)

- Multiple methods to fix SBOM information

  Fix license findings or project metadata such as source code repository/paths, artifacts URLs.

Collected Information

- Package name
- Version
- Source code repository URL
- Source and binary artifacts
- Copyright owner
- License and License URL
- Attribution and other notices and URLs
- List / tree of dependencies
- Location where the source code will be maintained

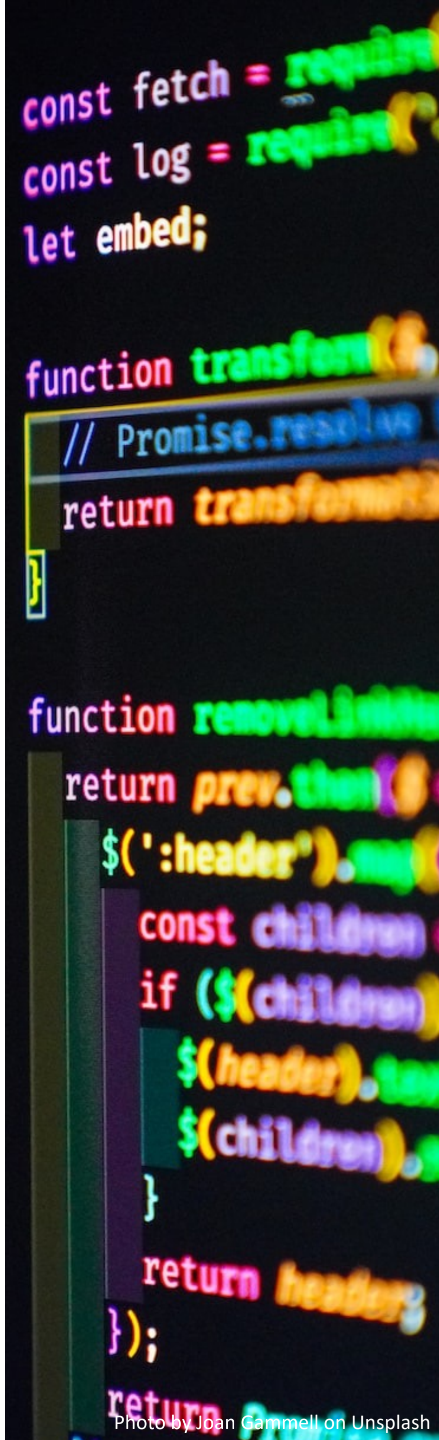
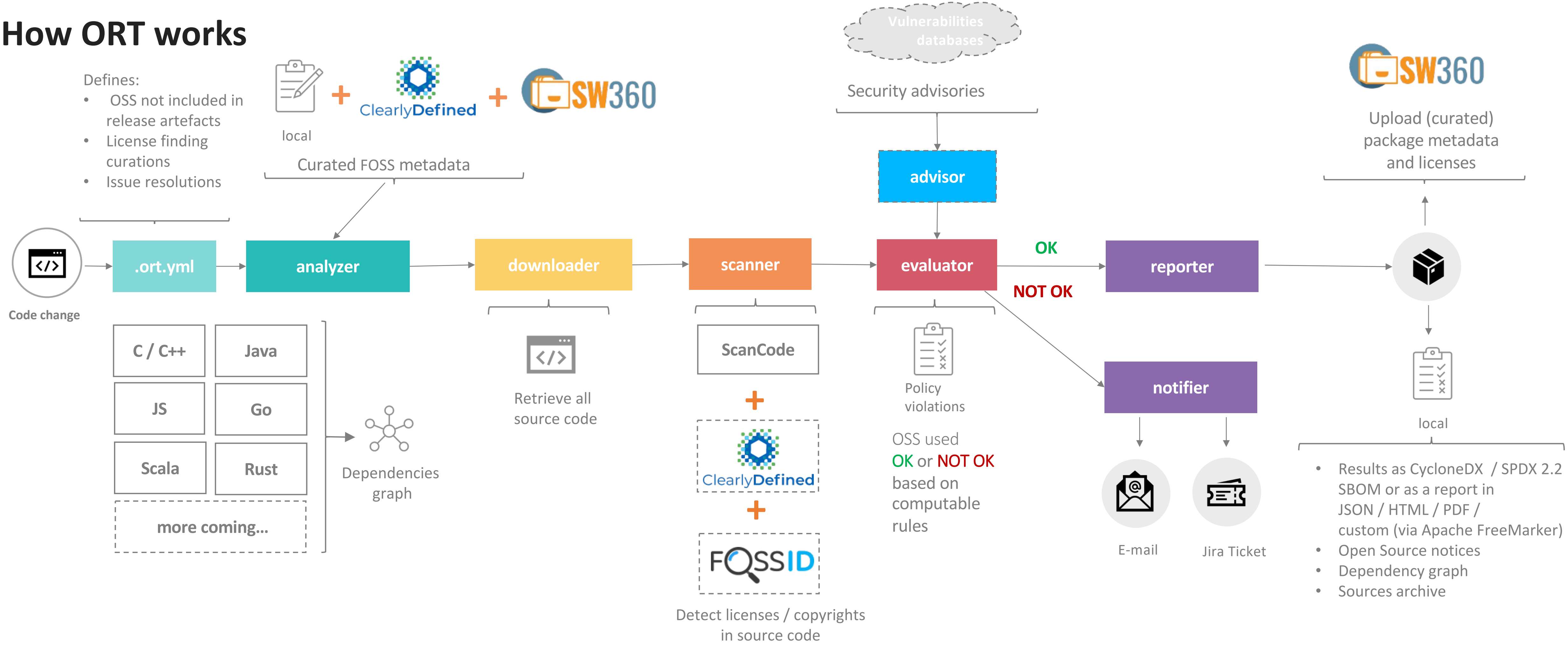


Photo by Joan Gammell on Unsplash

How ORT works



Defines:

- OSS not included in release artefacts
- License finding curations
- Issue resolutions

local + ClearlyDefined + SW360

Curated FOSS metadata

Vulnerabilities databases

Security advisories

advisor

SW360

Upload (curated) package metadata and licenses

Code change → .ort.yml → analyzer → downloader → scanner → evaluator → OK → reporter

NOT OK → notifier

C / C++ | Java | JS | Go | Scala | Rust | more coming...

Dependencies graph

Retrieve all source code

ScanCode + ClearlyDefined + FOSSID

Detect licenses / copyrights in source code

Policy violations

OSS used OK or NOT OK based on computable rules

E-mail

Jira Ticket

local

- Results as CycloneDX / SPDX 2.2 SBOM or as a report in JSON / HTML / PDF / custom (via Apache FreeMarker)
- Open Source notices
- Dependency graph
- Sources archive

OK/NOT OK = code context + legal context + product context + security context + ...

To use or publish Open Source

code context:

Source code, docs, example, test or build tools?

How is it included? Which scope? Linking?

Did we change the code?

legal context:

What are the licenses and resulting obligations?

Patents? Freedom to operate?

Created by us or FOSS community?

product context:

What is released to customers? Artifact, service or website?

What does the contract say?

security context:

What are the security advisories for open source packages included in the project?

What the health of community for the open source used?

Join the ORT community at github.com/oss-review-toolkit/ort

OSS
Review Toolkit

Demo

https://youtu.be/mqoW9sfTrqw

Photo by John Schnobrich on Unsplash

Questions?
Interested in collaborating?

Chat with us using the ort-talk Slack channel

github.com/oss-review-toolkit/ort

Thomas Steenbergen

github.com/tsteenbe

@tsteenbe

linkedin.com/in/tsteenbe

Photo by Cytonn Photography on Unsplash