FOSDEM'21

Software Composition Dev Room

Automating your license compliance policy with

OSS Review Toolkit

Photo by Dimitri Houtteman on Unsplash

About me

@tsteenbe
linkedin.com/in/tsteenbe

Head of Open Source

- HERE Open Source Office (OSO) is a team of 7 people
- Supporting 9k+ employees in 56 countries on all things Open Source together with our legal counsels

Active contributor to:

OSS Review Toolkit
http://oss-review-toolkit.org

SPDX

Open Source Tooling Group

OPENCHAIN

ClearlyDefined

TODO
European Chapter

How to automate?

Counsel: Found OSS with Apache-2.0, BSD-3-Clause, CC-BY-SA-3.0 and GPL-2.0 licenses.

Apache-2.0 and GPL-2.0 are incompatible with each other.

Please explain…

Engineer: Our code includes BSD-3-Clause and we depend on Apache-2.0 test library.

GPL-2.0 is build tools and CC-BY-SA-3.0 is docs from StackOverflow

Counsel: So what is distributed to our customers?

Engineer: An executable with only our code and BSD-3-Clause

Counsel: OK, but you must include a notices file in your release to comply with BSD-3-Clause license

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

code context	legal context	product context
Source code, docs, example, test or build tools?	**What are the licenses and resulting obligations?**	**What is released to customers? Artifact, service or website?**
How is it included? Which scope? Linking?	**Patents? Freedom to operate?**	**What does the contract say?**
Did we change the code?	**Created by us or FOSS community?**	

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

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

How is it included? Which scope? Linking?

Did we change the code?

What are the licenses and resulting obligations?

Patents? Freedom to operate?

Created by us or FOSS community?

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

What does the contract say?

Excludes
(.ort.yml)

\+

Curations
fix local findings
(.ort.yml)

Policy Rules
(rules.kts + license-classifications.yml)

\+

Curations
fix metadata/findings
(curations.yml)

Policy Rules
(rules.kts)

\+

Labels
ORT's CLI arguments

The code context (.ort.yml)

Included in the code repository to be scanned.

```
excludes:
  paths:
  - pattern: "*/src/{funTest,test}/**"
    reason: "TEST_OF"
    comment: "Licenses contained in this directory are used for testing and do not apply to the OSS Review Toolkit."
  scopes:
  - pattern: "test"
    reason: "TEST_DEPENDENCY_OF"
    comment: "Packages for testing only."
curations:
  license_findings:
  - path: "README.md"
    line_count: 1
    detected_license: "GPL-1.0-or-later"
    concluded_license: "NONE"
    reason: "DOCUMENTATION_OF"
    comment: "Findings reference a file with 'gpl' in its name."
```

Excludes - mark files, directories or package manager scopes as not included in released artifacts.

License finding curations - overwrite scan results to correct identified licenses.

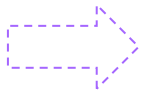
See a full example at github.com/oss-review-toolkit/ort/.ort.yml

The license context (license-classifications.yml)

Classify licenses in categories used in your policy (rules.kts). Used in all scans, part of ORT configuration files.

```
categories:
- name: "copyleft"
- name: "strong-copyleft"
- name: "copyleft-limited"
- name: "permissive"
  description: "Licenses with permissive obligations."
- name: "public-domain"
- name: "include-in-notice-file"
- name: "include-source-code-offer-in-notice-file"
categorizations:
- id: "CC0-1.0"
  categories:
  - "public-domain"
- id: "GPL-2.0-only WITH Classpath-exception-2.0"
  categories:
  - "copyleft-limited"
  - "include-in-notice-file"
  - "include-source-code-offer-in-notice-file"
```

Categories of licenses plus optional description



Assign license to a category

See a full example at github.com/oss-review-toolkit/ort/examples/license-classifications.yml

The license context (curations.yml file)

Patch package metadata. Used in all scans, part of ORT configuration files.

```
- id: "Maven:asm:asm" # No version means the curation will be applied to all versions of the package.
  curations:
    comment: "Repository moved to https://gitlab.ow2.org."
    vcs:
      type: "git"
      url: https://gitlab.ow2.org/asm/asm.git
- id: "NPM::ramda:[0.21.0,0.25.0]" # Ivy-style version matchers are supported.
  curations:
    comment: >-
      The package is licensed under MIT per `LICENSE` and `dist/ramda.js`. The project logo is CC-BY-NC-SA-3.0 but it is
      not part of the distributed .tar.gz package, see the `README.md` which says:
      "Ramda logo artwork © 2014 J. C. Phillipps. Licensed Creative Commons CC BY-NC-SA 3.0."
    concluded_license: "MIT"
- id: "PyPI::pyramid_workflow:1.0.0"
  curations:
    comment: "The package has an unmappable declared license entry."
    declared_license_mapping:
      "BSD-derived (http://www.repoze.org/LICENSE.txt)": "LicenseRef-scancode-repoze"
```

Update source code repository

Define effective license

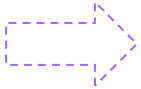
Map declared license to SPDX id

See a full example at github.com/oss-review-toolkit/ort/examples/curations.yml

The product context (labels)

Pass any labels you want as command-line arguments when running ORT.

```
cli/build/install/ort/bin/ort analyze \
    --package-curations-file $ORT_CONFIG_DIR/curations.yml \
    --clearly-defined-curations
    -P ort.analyzer.allowDynamicVersions=true \
    -i $ANALYZER_INPUT_DIR \
    -o $ ANALYZER_OUTPUT_DIR \
    -f JSON \
    -l project="company-proprietary"
    -l dist="external"
    -l org="engineering-usa-boston"
    -l PROD_ID="SK-M2020W/EU" \
    -l REVIEW_ID="1268" \
    -l JIRA_ID="SK-7891" \
    -l SW_NAME="Example Product Component AX" \
    -l SW_VERSION="1.06"
```



Use `-l` to pass your set of labels to pass additional information that's useful to scan report readers or to use in your policy rules

Your custom policy (rules.kts file)

You can write rules using all data ORT captures including

- Package metadata
- Declared, detected and concluded licenses
- Labels
- Security vulnerabilities (coming soon)

Rules are written in Kotlin script and passed as a parameter to ORT's Evaluator component

```
cli/build/install/ort/bin/ort evaluate \
  -i $SCANNER_OUTPUT_DIR/scan-result.yml
  -o $EVALUATOR_OUTPUT_DIR \
  --output-formats YAML \
  --license-classifications-file $ORT_CONFIG_DIR/license-classifications.yml \
  --package-curations-file $ORT_CONFIG_DIR/curations.yml  \
  --rules-file $ORT_CONFIG_DIR/rules.kts
```


See the docs at github.com/oss-review-toolkit/ort/blob/master/docs/file-rules-kts.md

The rules.kts file - Import classifications

Import from license-classifications.yml

```
fun getLicenseCategory(categoryId: String) =
    licenseClassifications.getLicensesForCategory(categoryId).map { it.id }.toSet()

val permissiveLicenses = getLicenseCategory("permissive")

val copyleftLicenses = getLicenseCategory("copyleft")

val copyleftLimitedLicenses = getLicenseCategory("copyleft-limited")

val publicDomainLicenses = getLicenseCategory("public-domain")
```


See a full example at github.com/oss-review-toolkit/ort/examples/rules.kts

The rules.kts file – Rule Types

- PackageRule
  - Executed once for each found package
- DependencyRule
  - Executed for each dependency
- LicenseRule
  - Executed within a PackageRule or DependencyRule
    for each license found in a user specified set of licenses (LicenseView*)

*See for available options github.com/oss-review-toolkit/ort/blob/master/model/src/main/kotlin/licenses/LicenseView.kt

The rules.kts file - LicenseRule

Create LicenseRule for each license category so you can use them in your rules

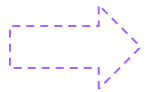
```
fun PackageRule.LicenseRule.isCopyleft() =
    object : RuleMatcher {
        override val description = "isCopyleft($license)"

        override fun matches() = license in copyleftLicenses
    }
```

Match "copyleft" category from license-classifications.yml

```
fun PackageRule.LicenseRule.isCopyleftLimited() =
    object : RuleMatcher {
        override val description = "isCopyleftLimited($license)"

        override fun matches() = license in copyleftLimitedLicenses
    }
```



Match "copyleft-limited" category from license-classifications.yml

See a full example at github.com/oss-review-toolkit/ort/examples/rules.kts

The rules.kts file - PackageRule

```
// Define a rule that is executed for each package.
packageRule("COPYLEFT_IN_SOURCE") {
        require {
            // Do not trigger this rule on packages that have been excluded in the .ort.yml.
            -isExcluded()
            +labelContains("project", "company-proprietary")
        }
        // Define a rule that is executed for each license of the package.
        licenseRule("COPYLEFT_IN_SOURCE", LicenseView.CONCLUDED_OR_DECLARED_AND_DETECTED) {
            require {
                -isExcluded()
                +isCopyleft()
            }
         // Throw an error message including guidance how to fix the issue.
         error(
             "The $license was ${licenseSource.name.toLowerCase()} in package ${pkg.id.toCoordinates()}.”,
             "A text written in MarkDown to help users resolve policy violations which may link to additional resources."
         )
}
```

See a full example at github.com/oss-review-toolkit/ort/examples/rules.kts

The rules.kts file - DependencyRule

```
// Define a rule that is executed for each dependency of a project.
dependencyRule("COPYLEFT_LIMITED_STATIC_LINK_IN_DIRECT_DEPENDENCY") {
        require {
            +isAtTreeLevel(1)
            +isStaticallyLinked()
        }

        licenseRule("LINKED_WEAK_COPYLEFT", LicenseView.CONCLUDED_OR_DECLARED_OR_DETECTED) {
            require {
                +isCopyleftLimited()
            }

            // Use issue() instead of error() if you want to set the severity.
            issue(
                Severity.WARNING,
                "The project ${project.id.toCoordinates()} has a statically linked direct dependency licensed under the $license.",
                 "A text written in MarkDown to help users resolve policy violations which may link to additional resources.”
           )
```

See a full example at github.com/oss-review-toolkit/ort/examples/rules.kts

The rules.kts file – Built-in Rule Operators

Rule:

- hasLabel(string) - check if label with provided string exists
- labelContains(string1, string2) - check if label provided string1 exists with value string2

PackageRule:

- hasLicense - check if package has any concluded, declared or detect license
- isExcluded - check if package has been excluded via .ort.yml
- isFromOrg(string[]) - check if package id matches in provided string array
- isMetaDataOnly - check if package only contain metadata
- isSpdxLicense - check if license is an official SPDX identifier
- isType(string) - check if package type matches provided string

DependencyRule:

- isAtTreeLevel(int) - check if dependency is in dependency tree at provided integer level
- isProjectFromOrg(string[]) - check if project id matches in provided string array
- isStaticallyLinked(string) - check if dependency is statically linked to its parent

See also github.com/oss-review-toolkit/ort/blob/master/evaluator/src/main/kotlin/{Rule.kt|PackageRule.kt|DependencyRule.kt}

Thank you
Merci
Dank je wel

Thomas Steenbergen

@tsteenbe

linkedin.com/in/tsteenbe

OSS Review Toolkit

https://github.com/oss-review-toolkit/ort

Starting and Scaling an Open Source Office

ORT Slack

Related OSS Projects

https://oss-compliance-tooling.org

https://clearlydefined.io

https://spdx.org

https://www.openchainproject.org

https://www.doubleopen.org

https://www.eclipse.org/sw360