



Scancode evaluation

1	Contents	
2	Scancode installation	3
3	Scancode output format	3
4	Select Apertis packages to evaluate scancode	4
5	Run scan-copyrights as gold standard	4
6	Run scancode	5
7	Analysis time	6
8	Analysis time with <code>-processes</code> from 1 to 8	7
9	Analysis time with <code>-timeout X</code>	8
10	Analysis time with <code>-max-in-memory 0</code>	9
11	Analysis time with <code>ONLY -license</code>	11
12	Reliability of detected license	11
13	No license deduction for project and folder	12
14	Statistic about files without detected license	13
15	DEP5 invalid format	14
16	scancode output format	14
17	Summary	15
18	Required resources for analysis	15
19	License scan accuracy	15
20	Output format	15
21	Outdated debian/apertis/copyright.yaml	15
22	Proposed plan	16
23	References	17
24	Currently, scan-copyrights ¹ (which uses licensecheck ² under the hood) is used in	
25	Apertis to scan copyright/license notices. This tool has some downsides, thus we	
26	are evaluating to use scancode-toolkit ³ instead. A comparison of <code>licensecheck</code>	
27	vs <code>scancode</code> is available on the ScanCode's website ⁴ , TL;DR: <i>scancode is more</i>	
28	<i>accurate but slower</i> .	
	¹ https://tracker.debian.org/pkg/libconfig-model-dpkg-perl	
	² https://tracker.debian.org/pkg/licensecheck	
	³ https://github.com/nexB/scancode-toolkit	
	⁴ https://scancode-toolkit.readthedocs.io/en/stable/misc/faq.html#how-is-scancode-different-from-debian-licensecheck	

29 `scancode-toolkit` has an option to export results as [DEP5 format](#)⁵ (see
30 [GH#472](#)⁶) which is the format currently use by Apertis license tooling. That
31 means, `scancode-toolkit` is potentially compatible with the rest of the Apertis
32 licensing tooling.

33 Scancode installation

34 `scancode` is not available as Debian package ([GH#1580](#)⁷ and [GH#3253](#)⁸) nor
35 as a Docker image ([GH#3026](#)⁹), but a [Dockerfile](#)¹⁰ is provided by upstream¹¹.
36 That means, we can create our own Docker image, or we can reuse the OSS
37 Review Toolkit Docker image which integrates `scancode`. Since the ORT Docker
38 image used in our pipeline is outdated, it would be easier for now to decou-
39 ple `scancode` from the ORT docker image to avoid having to use an outdated
40 `scancode` (`scancode` used in the ORT image is [one year old](#)¹²).

41 Here are the steps to build a docker image:

```
1  git clone https://github.com/nexB/scancode-toolkit
2  cd scancode-toolkit
3  LATEST_VER=v32.0.8
4  git checkout $LATEST_VER
5  docker build --tag scancode-toolkit --tag scancode-toolkit:$LATEST_VER .
```

42 Scancode output format

43 Scancode is able to write its output in different formats:

```
44 docker run  scancode-toolkit --help
45 ...
46 output formats:
47     --json FILE           Write scan output as compact JSON to FILE.
48     --json-pp FILE       Write scan output as pretty-printed JSON to FILE.
49     --json-lines FILE     Write scan output as JSON Lines to FILE.
50     --yaml FILE           Write scan output as YAML to FILE.
51     --csv FILE            [DEPRECATED] Write scan output as CSV to FILE. The
52                          --csv option is deprecated and will be replaced by
```

⁵<https://www.debian.org/doc/packaging-manuals/copyright-format/1.0/>

⁶<https://github.com/nexB/scancode-toolkit/issues/472>

⁷<https://github.com/nexB/scancode-toolkit/issues/1580>

⁸<https://github.com/nexB/scancode-toolkit/issues/3253>

⁹<https://github.com/nexB/scancode-toolkit/issues/3026>

¹⁰<https://github.com/nexB/scancode-toolkit/blob/develop/Dockerfile>

¹¹<https://scancode-toolkit.readthedocs.io/en/latest/getting-started/install.html#installat-ion-via-docker>

¹²<https://github.com/nexB/scancode-toolkit/releases/tag/v31.2.4>

```

53         new CSV and tabular output formats in the next
54         ScanCode release. Visit
55         https://github.com/nexB/scancode-toolkit/issues/3043
56         to provide inputs and feedback.
57     --html FILE           Write scan output as HTML to FILE.
58     --custom-output FILE Write scan output to FILE formatted with the custom
59                           Jinja template file.
60     --debian FILE        Write scan output in machine-readable Debian
61                           copyright format to FILE.
62     --custom-template FILE Use this Jinja template FILE as a custom template.
63     --cyclonedx FILE      Write scan output in CycloneDX JSON format to FILE.
64     --cyclonedx-xml FILE  Write scan output in CycloneDX XML format to FILE.
65     --spdx-rdf FILE       Write scan output as SPDX RDF to FILE.
66     --spdx-tv FILE        Write scan output as SPDX Tag/Value to FILE.
67     ...

```

68 These formats include:

- 69 • Debian DEP5 format, the one already in use with Apertis licensing tooling.
- 70 • YAML, widely used in Apertis and used to teach `scan-copyrights` detection
- 71 license issues (i.e. `debian/apertis/copyright.yml`).
- 72 • SPDX, open standard for communicating SBOM information.
- 73 • CycloneDX, another SBOM standard.

74 Initially, it should be simpler to continue using the Debian DEP5 since the whole
75 Apertis licensing tooling is using it. But for a long term plan, we may want to
76 switch to a more widely used format like `SPDX` or `CycloneDX` which are also com-
77 patible with `ORT` and other tools. This should make the Apertis license/SBOM
78 processes more flexible.

79 Select Apertis packages to evaluate scancode

80 Let's use packages that are wrongly detected by `scan-copyrights` (i.e. packages
81 with the use of `override-license` in `debian/apertis/copyright.yml`).

82 Here is a small random list of packages based on a local `grep` of `override-license:`
83 `debianutils`, `libarchive`, `libgdata`, `libunistring`, `libusb`, `nss`, `nss-pem`, `openjpeg2`,
84 `openssl` `xorg-server`.

85 Run scan-copyrights as gold standard

86 First, `scan-copyrights` is run on the package in a `v2025dev2` VM:

```
1  # Because of the use of "override-license" in debian/apertis/copyright.yml
2  LIST_PKGS=" debianutils libarchive libgdata libunistring libusb nss nss-pem openjpeg2 openssl xorg-servers
3  # Adding other well known pkgs
4  LIST_PKGS+=" pipewire rust-coreutils "
5  for PKG in $LIST_PKGS:
6  do
7      git clone https://gitlab.apertis.org/pkg/${PKG}.git
8      cd ${PKG}
9      /usr/bin/time -f "%e" scan-copyrights > ../${PKG}-scan-copyright 2> ../${PKG}-time
10     cd ..
11 done
```

87 Run scancode

88 Now, scancode is run by excluding `debian/copyright` and `debian/apertis/copyright`
89 because they can easily confuse scancode (see [GH#2885¹³](https://github.com/nexB/scancode-toolkit/issues/2885#issuecomment-1136268172)).

¹³<https://github.com/nexB/scancode-toolkit/issues/2885#issuecomment-1136268172>

```

1  # Because of the use of "override-license" in debian/apertis/copyright.yml
2  LIST_PKGS=" debianutils libarchive libgdata libunistring libusb nss nss-pem openjpeg2 openssl xorg-ser
3  # Adding other well known pkgs
4  LIST_PKGS+=" pipewire rust-coreutils firefox-esr"
5  for PKG in $LIST_PKGS:
6  do
7      git clone https://gitlab.apertis.org/pkg/${PKG}.git
8      cd ${PKG}
9
10     # DEP5 output
11     docker run -v $PWD:/project scancode-toolkit \
12         --copyright --license --license-text --strip-root \
13         --ignore */debian/copyright --ignore */debian/apertis/copyright \
14         --ignore */debian/apertis/${PKG}-scancode-copyright \
15         -n 8 --debian /project/debian/apertis/${PKG}-scancode-copyright \
16         /project/.
17
18     # YAML output
19     docker run -v $PWD:/project scancode-toolkit \
20         --copyright --license --license-text --strip-root \
21         --ignore */debian/copyright --ignore */debian/apertis/copyright \
22         --ignore */debian/apertis/${PKG}-scancode-copyright \
23         -n 8 \
24         --yaml /project/debian/apertis/${PKG}-scancode-copyright-yaml \
25         /project/.
26
27     cd ..
28 done

```

90 Analysis time

91 This analysis was performed on a XPS13-9310 laptop with a CPU i7-1185G7
92 (@3.00GHz×8), 16 GB RAM and an SSD hard disk.

Package	Time scan-copyrights	Time scancode	Diff
debianutils	1.3 s	38 s	~29 times slower
libarchive	6.6 s	7 m 25 s	~67 times slower
libgdata	4.7 s	3 m 55 s	~50 times slower
libunistring	8.1 s	10 m 48 s	~80 times slower
libusb	1.2 s	54 s	~45 times slower
nss	25.8 s	28 m 47 s	~67 times slower
nss-pem	28.6 s	26 m 59 s	~56 times slower
openjpeg2	3.7 s	3 m 8 s	~50 times slower

Package	Time scan-copyrights	Time scancode	Diff
openssl	23.6 s	18 m 3 s	~46 times slower
pipewire	6.1 s	4 m 57 s	~48 times slower
rust-coreutils	4.4 s	1 m 54 s	~26 times slower
xorg-server	9.1 s	9 m 24 s	~62 times slower
firefox-esr*	XX s	OOM killed after ~ 1 d [1]	~XX times slower

- `firefox-esr` is one of the biggest packages in Apertis, but is not in the target repository. Thus, we wouldn't have to analyze it with `scancode`, but it is used here to evaluate `scancode` in the worst cases.
- [1] `scancode` ran on `firefox-esr` for ~ 23 hours and 30 mins before being OOM killed. It seems, the scan was over and `scancode` was processing data to generate its output file when it was killed. Its scanning parallel processes have stopped, only the main process was running and the used RAM was at ~ 3 GB (of 16 GB available) about 10 mins before OOM.
- While doing the analysis with 8 parallel processes, all of them were at 100% during the entire analysis time, so at least the CPU is a bottleneck.

Analysis time with `--processes` from 1 to 8

From the [scancode options](#)¹⁴:

```
-n, --processes INTEGER
    Scan <input> using n parallel processes. [Default: 1]
```

This option allows to use several processes for scanning files.

```
1  PKG="debianutils"
2  git clone https://gitlab.apertis.org/pkg/${PKG}.git
3  cd ${PKG}
4
5  for N in {1..8}
6  do
7      # YAML output
8      docker run -v $PWD:/project scancode-toolkit \
9          --copyright --license --license-text --strip-root \
10         --ignore */debian/copyright --ignore */debian/apertis/* \
11         -n ${N} \
12         --yaml /project/debian/apertis/${PKG}-${N}-scancode-copyright-yaml \
13         /project/.
14  done
```

¹⁴<https://scancode-toolkit.readthedocs.io/en/stable/cli-reference/core-options.html>

N processes	Time scandcode —debianutils
1	1 m 34.5 s
2	58.2 s
3	45.4 s
4	39.0 s
5	38.1 s
6	37.2 s
7	36.7 s
8	36.0 s

109 Adding more parallel processes improve the scanning time, but it seems we are
110 reaching a threshold at ~ 4 parallel processes where adding more processes only
111 slightly improves the scanning time. This may be due to the fact that the tested
112 package is *small*. For a bigger package like `firefox-esr`, this threshold may be
113 higher and it could be beneficial to have more parallel processes.

114 Analysis time with `-timeout X`

115 From the [scancode options](#)¹⁵:

116 `--timeout FLOAT`
117
118 Stop scanning a file if scanning takes longer than a timeout in seconds. [Default: 120]

119 This option allows to avoid getting stuck on a file for too long.

```

1  PKG="debianutils"
2  git clone https://gitlab.apertis.org/pkg/${PKG}.git
3  cd ${PKG}
4
5  for N in 120 110 100 90 80 70 60 30 10
6  do
7      # YAML output
8      docker run -v $PWD:/project scancode-toolkit \
9          --copyright --license --license-text --strip-root \
10         --ignore */debian/copyright --ignore */debian/apertis/* \
11         --timeout ${N} -n 8 \
12         --yaml /project/debian/apertis/${PKG}-${N}-scancode-copyright-yaml \
13         /project/.
14 done
```

¹⁵<https://scancode-toolkit.readthedocs.io/en/stable/cli-reference/core-options.html>

Timeout	Time scancode -n 8 —debianutils
120 (default)	38.5 s
110	43.6 s
100	44.1 s
90	39.9 s
80	40.3 s
70	38.3 s
60	37.8 s
30	37.9 s
10	29.6 s

120 Decreasing the timeout per file seems to be quite efficient to reduce the scanning
121 time, but since some files are no longer fully scanned, a more comprehensive
122 comparison of detected licenses should be done to ensure we are not losing too
123 much data.

```

1  PKG="firefox-esr"
2  git clone https://gitlab.apertis.org/pkg/${PKG}.git
3  cd ${PKG}
4
5  date
6  docker run -v $PWD:/project scancode-toolkit \
7      --copyright --license --license-text --strip-root \
8      --ignore */debian/copyright --ignore */debian/apertis/* \
9      --timeout 10 -n 8 \
10     --yaml /project/debian/apertis/${PKG}-scancode-timeout-10-copyright-yaml \
11     /project/.
12  date

```

124 Interestingly, the `--timeout 10` option on the `firefox-esr` scan avoids triggering
125 the OOM kill issue, probably because more files are no longer scanned, thus `scan-`
126 `code` has to manage a smaller dataset and therefore a smaller memory footprint.
127 Unfortunately, this option is not enough to make scanning time acceptable since
128 it took 35 hours to complete the full scan of `firefox-esr`.

129 Analysis time with `--max-in-memory 0`

130 From the [scancode options](#)¹⁶:

```

131 --max-in-memory INTEGER
132
133     Maximum number of files and directories scan details kept in memory during a
134     scan. Additional files and directories scan details above this number are

```

¹⁶<https://scancode-toolkit.readthedocs.io/en/stable/cli-reference/core-options.html>

cached on-disk rather than in memory. Use 0 to use unlimited memory and
 disable on-disk caching. Use -1 to use only on-disk caching. [Default: 10000]

Based on an upstream issue (see [GH#1014¹⁷](#)), the disk cache seems to be really
 slow.

```

1 # Because of the use of "override-license" in debian/apertis/copyright.yml
2 LIST_PKGS=" debianutils libarchive libgdata libunistring libusb nss nss-pem openjpeg2 openssl xorg-ser
3 # Adding other well known pkgs
4 LIST_PKGS+=" pipewire rust-coreutils firefox-esr"
5 for PKG in $LIST_PKGS:
6 do
7     git clone https://gitlab.apertis.org/pkg/${PKG}.git
8     cd ${PKG}
9     # YAML output
10    docker run -v $PWD:/project scancode-toolkit \
11        --copyright --license --license-text --strip-root \
12        --ignore */debian/copyright --ignore */debian/apertis/* \
13        --max-in-memory 0 -n 8 \
14        --yaml /project/debian/apertis/${PKG}-scancode-copyright-yaml \
15        /project/.
16    cd ..
17 done

```

Package	Time scan-copyrights	Time scancode	Diff
debianutils	1.3 s	34 s	~26 times slower
libarchive	6.6 s	6 m 41 s	~60 times slower
libgdata	4.7 s	3 m 35 s	~46 times slower
libunistring	8.1 s	11 m 12 s	~83 times slower
libusb	1.2 s	54 s	~45 times slower
nss	25.8 s	27 m 47 s	~66 times slower
nss-pem	28.6 s	26 m 49 s	~57 times slower
openjpeg2	3.7 s	3 m 14 s	~52 times slower
openssl	23.6 s	18 m	~46 times slower
pipewire	6.1 s	5 m 29 s	~54 times slower
rust-coreutils	4.4 s	2 m 3 s	~28 times slower
xorg-server	9.1 s	10 m 9 s	~67 times slower
firefox-esr*	XXX s	OOM killed after ~ 1 d [1]	~XX times slower

Passing `--max-in-memory 0` to `scancode` doesn't improve scanning time since these
 results are of the same order of magnitude (+/- random fluctuation) to the ones
 without this option.

¹⁷<https://github.com/nexB/scancode-toolkit/issues/1014>

142 Analysis time with ONLY -license

143 i.e. without --copyright --license-text

```

1  PKG="nss"
2  git clone https://gitlab.apertis.org/pkg/${PKG}.git
3  cd ${PKG}
4
5  # YAML output
6  docker run -v $PWD:/project scancode-toolkit \
7      --license --strip-root \
8      --ignore */debian/copyright --ignore */debian/apertis/* \
9      -n 8 \
10     --yaml /project/debian/apertis/${PKG}-scancode-ONLYlicense-copyright-yaml \
11     /project/.

```

Package	Time scan-copyrights	Time scancode with copyright	Time scancode without copyright	Improvement
nss	25.8 s	27 m 47 s	21 m 42 s	22%

144 Avoiding the scan for copyrights and only checking licenses slightly decreases
145 the scanning time but it remains in the same order of magnitude without signif-
146 icantly improving the comparison with scan-copyright.

147 Reliability of detected license

148 *Some of debian/apertis/copyright.yaml files used are no longer required since the*
149 *Bookworm rebase, so all packages analyzed don't have a problematic file which*
150 *can be used to compare scan-copyrights and scancode.*

Package	File	Actual license
libarchive	shar.1 ¹⁸	BSD-4-Clause-UC
libgdata	README ¹⁹	LGPL-2.1-or-later
libunistring	version.c ²⁰	LGPL-3.0-or-later OR GPL-2.0-or-later
libusb	06_bsd.diff ²¹	BSD-2-Clause [1]
nss	derdump.1 ²²	MPL-2.0 [2]
nss-pem	doc/rst/legacy/ ^{*23}	MPL-1.1 OR GPL-2.0-only OR LGPL-2.1-only
openjpeg2	opj_getopt.c ²⁴	BSD-3-Clause
openssl	cmll-x86*.pl ²⁵	Apache-2.0 OR GPL-2.0-or-later OR LGPL-2.1-or-later OR MPL-2.0
xorg-server	hw/xwin/winprefsyacc.* ²⁶	GPL-3.0-or-later WITH Bison-exception-2.2 AND LicenseRef-scancode

¹⁸<https://gitlab.apertis.org/pkg/libarchive/-/blob/13653b9a562c658133b37e5859693898a7b929e8/contrib/shar/shar.1>

- [0] scan-copyrights has improved in Bookworm.
- [1] Retrospective change: <https://www.netbsd.org/about/redistribution.html#why2clause>
- [1.1] BSD-2-Clause-NetBSD and/or BSD-2-clause and/or BSD-3-clause and/or FSFUL and/or FSFULLR and/or GPL-2 and/or LGPL-2 and/or X11
- [2] Simplified upstream with bookworm, debian/apertis/copyright.yaml outdated.
- [3] A second license is later in the code

scancode is better to deal with complex licenses combinations, especially because it scans the whole file and not only the first lines. Moreover, it reports all licenses detected with a matching score (available in the YAML output).

No license deduction for project and folder

While scan-copyrights is able to perform some deduction of license/copyright for a project and/or folder, scancode only performs scanning at file level.

For instance, scan-copyrights gives the following result for openssl:

```
Files: *
Copyright: 1998-2023, The OpenSSL Project
          1995-1998, Eric A. Young, Tim J. Hudson
License: Apache-2.0

...

Files: crypto/ec/asm/*
Copyright: 1998-2023, The OpenSSL Project Authors.
License: Apache-2.0 and/or OpenSSL
```

This result give us the information that the project is under the license Apache-2.0 and files in crypto/ec/asm/* are under Apache-2.0 and/or OpenSSL licenses.

¹⁹<https://gitlab.apertis.org/pkg/libgdata/-/blob/6e88c1a214f86c3025b0124a97b557c3710df339/README>

²⁰<https://gitlab.apertis.org/pkg/libunistring/-/blob/d4765b859b6ca17f0ea885d3ec9adea691dc1133/lib/version.c>

²¹https://gitlab.apertis.org/pkg/libusb/-/blob/5890eb277e80734cea681e58c52712a06516d85b/debian/patches/06_bsd.diff

²²<https://gitlab.apertis.org/pkg/nss/-/blob/4a93da238116e085ad1d949b650e9ec32d98038a/nss/doc/nroff/derdump.1>

²³https://gitlab.apertis.org/pkg/nss-pem/-/blob/65192a7e89364a15fadb7b2c832a72a0efa16e63/nss/nss/doc/rst/legacy/nss_3.12.2_release_notes.html/index.rst

²⁴https://gitlab.apertis.org/pkg/openjpeg2/-/blob/a1d3412b2721a1d7dbd7cd3d7f13bd6647bda7f5/src/bin/common/opj_getopt.c

²⁵<https://gitlab.apertis.org/pkg/openssl/-/blob/f603f6ae103b26ef89089fc63945e833c0401839/crypto/camellia/asm/cml1-x86.pl>

²⁶<https://gitlab.apertis.org/pkg/xorg-server/-/blob/2d6db144d7c3393d2f3f113168b4f421192c90c8/hw/xwin/winprefsyacc.c>

179 This behavior allows to assign a license to files by inheriting it from the license
180 of the project (or from the higher level folder's license).

181 Some projects don't add license/copyright information in all of their files, which
182 could be annoying for `scancode` as it won't be able to detect the right license.
183 We would need to add this logic in `scancode` or in another Apertis script (like
184 `ci-license-scan`²⁷?).

185 Some related upstream issues:

- 186 • [Proposal: Scan deduction and summarization](#)²⁸
- 187 • [Primary license detections not shown properly in `debian\_copyright`](#)²⁹

188 **Statistic about files without detected license**

189 The yaml file generated by `scancode` is sometimes malformed due to `--license-`
190 `text` (see [#GH3219](#)³⁰, but seems not enough).

```
1  # Because of the use of "override-license" in debian/apertis/copyright.yml
2  LIST_PKGS=" debianutils libarchive libgdata libunistring libusb nss nss-pem openjpeg2 openssl xorg-ser
3  # Adding other well known pkgs
4  LIST_PKGS+=" pipewire rust-coreutils firefox-esr"
5  for PKG in $LIST_PKGS:
6  do
7      git clone https://gitlab.apertis.org/pkg/${PKG}.git
8      cd ${PKG}
9
10     # YAML output
11     docker run -v $PWD:/project scancode-toolkit \
12         --license --strip-root \
13         --ignore */debian/copyright --ignore */debian/apertis/copyright \
14         --ignore */debian/apertis/${PKG}-scancode-copyright \
15         -n 8 \
16         --yaml /project/debian/apertis/${PKG}-scancode-copyright-yaml \
17         /project/.
18
19     cd ..
20 done
```

²⁷<https://gitlab.apertis.org/infrastructure/apertis-docker-images/-/blob/apertis/v2025dev/2/package-source-builder/overlay/usr/bin/ci-license-scan>

²⁸<https://github.com/nexB/scancode-toolkit/issues/377>

²⁹<https://github.com/nexB/scancode-toolkit/issues/3424>

³⁰<https://github.com/nexB/scancode-toolkit/issues/3219>

Package	Files number	Files number with license	Detected license %
debianutils	134	44	32.8 %
libarchive	1420	716	51.9 %
libgdata	705	326	46.2 %
libunistring	2118	1961	92.5 %
libusb	96	30	31.2 %
nss	4531	2393	52.8 %
nss-pem	4574	2423	52.9 %
openjpeg2	478	334	69.8 %
openssl	4655	3349	72 %
pipewire	1251	952	76 %
rust-coreutils	1311	326	24.8 %
xorg-server	1791	1227	68.5 %
firefox-esr*	XXX	XXX	XXX

DEP5 invalid format

`scancode` generates malformed `files stanza`. As defined in the [debian/copyright specification](#)³¹, each `files stanza` is composed by mandatory fields (i.e. `Files`, `Copyright` and `License`) and one optional field (i.e. `Comment`). For instance:

```
Files: Xext/sleepuntil.h
Copyright: 1993-2003, The XFree86 Project, Inc.
License: Expat
```

When `scancode` is not able to define a copyright or a license for a file, then it creates a stanza with only the `Files` field whereas `scan-copyrights` fills the missing field with `UNKNOWN`. Here is an example of malformed stanza by `scancode`:

```
Files: CODE_OF_CONDUCT.md
```

Here is another example from `scan-copyrights` where a missing field is filled:

```
Files: xkb/Makefile.in
Copyright: 1994-2021, Free Software Foundation, Inc.
License: UNKNOWN
```

This issue should easy be fixable in `scancode`, by filing missing field with an `UNKNOWN` value.

scancode output format

Support of `DEP5` format is incomplete in `scancode`, but bigger issues can probably be easily fixed.

³¹<https://www.debian.org/doc/packaging-manuals/copyright-format/1.0/#files-stanza>

211 YAML format gives way more information like: lines of the detected licenses,
212 the pattern, a score of matching, several identifiers of the licenses detected, etc.
213 Having all of these information may be useful for future enhancement of Apertis
214 license tooling.

215 Summary

216 Their different approaches in file analysis explains why `scancode` is slower, but
217 is able to detect way more licenses than `scan-copyrights` (see [upstream compar-](#)
218 [ison](#)³²):

- 219 • `licensecheck` is “a Perl script using hand-crafted regex patterns to find
220 typical copyright statements and about 50 common licenses”;
- 221 • `scancode`’s detection “is based on a (large) number of license full texts
222 (~2100) and license notices, mentions and variants (~32,000) and is data-
223 driven as opposed to regex-driven. It detects and reports exactly where
224 license text is found in a file. Just throw in more license texts to improve
225 the detection.”

226 Required resources for analysis

227 `scancode` is ~50 times slower than `scan-copyrights`. `scancode` requires much more
228 RAM than `scan-copyrights`.

229 License scan accuracy

- 230 • `scan-copyrights` has improved between Bullseye and Bookworm.
- 231 • `scancode` has a better detection for complex cases.

232 Output format

- 233 • `DEP5` incomplete support, but already used by apertis license tooling.
- 234 • YAML seems a sensible alternative since it provides many more information,
235 but would require to adapt apertis license tooling to this new format.

236 Outdated `debian/apertis/copyright.yaml`

237 This file would need to be refreshed in Apertis packages since the Bookworm
238 rebase. `scan-copyrights` is smarter and some packages have fixed their licensing
239 issues.

³²<https://scancode-toolkit.readthedocs.io/en/stable/misc/faq.html#how-is-scancode-different-from-debian-licensecheck>

240 Proposed plan

241 Some general guidelines:

- 242 • We need to come with a progressive approach, this is not something that
- 243 will happen from one day to the other
- 244 • Most of the packages are small and should take a reasonable amount of
- 245 time to scan
- 246 • We should be able to selectively disable scancode when necessary
- 247 • We can add additional logic to only scan the files that have changed since
- 248 last scan

249 Proposed plan to use scancode instead of scan-copyrights:

- 250 1. Update the docker image used to generate ORT reports in order to reuse
- 251 it for scancode. Apertis uses a handcrafted docker image to generate ORT
- 252 reports, since this image already contains scancode, it's possible to reuse
- 253 it to run scancode. The first step is to switch to an up-to-date [image](#)
- 254 [provided by ORT](#)³³. This step will require to adjust some scripts use by
- 255 Apertis including the template used to generate ORT reports.
- 256 2. Fix the DEP5 format created by scancode by adding missing mandatory
- 257 fields. Scancode generates report in the [DEP5 format](#)³⁴, unfortunately,
- 258 mandatory fields are missing when the copyright/license is not detected
- 259 (see [GH#3714](#)³⁵). Instead, scancode should fill missing field with UNKNOWN
- 260 or no-info-found as done by scan-copyrights.
- 261 3. Add support of “license deduction for project and folder” to scancode. scan-
- 262 code is unable to deduce a license for a whole project/folder based on the
- 263 license of other files. Without this feature, ~ 50% files will have a missing
- 264 license which is a regression compared to scan-copyrights. This task con-
- 265 sists in adding a logic to scancode to deduce a license for a folder and/or
- 266 project.
- 267 4. Add a new job running scancode to the ci-package-builder pipeline in
- 268 parallel to the current scan-licenses job using scan-copyrights.
- 269 5. Generate a new scancode report for all packages in target using the job
- 270 added in the previous step.
- 271 6. In the SBOM logic, add preference to use the scancode report if available
- 272 otherwise use the one from scan-copyrights.

273 Some other tasks can be done in parallel:

- 274 • Investigate how to use caching to avoid scanning files already scanned in
- 275 a previous run.
- 276 • Investigate how to improve performance of scancode (speed and RAM
- 277 usage).

³³<https://github.com/oss-review-toolkit/ort/pkgs/container/ort>

³⁴<https://www.debian.org/doc/packaging-manuals/copyright-format/1.0/>

³⁵<https://github.com/nexB/scancode-toolkit/issues/3714>

278 References

- 279 • [scancode-toolkit](#)³⁶

³⁶<https://github.com/nexB/scancode-toolkit>