Building on the Builds: Generating SBoMs from Yocto

Source code license forensics: The Yocto Project and SPDX

Kate Stewart, Senior Director of Strategic Programs
Richard Purdie, Yocto Project Architect

March 14, 2019

THE LINUX FOUNDATION

Building Embedded OS-Based Systems

THE LINUX FOUNDATION

What is the Yocto Project?

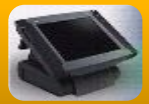
The Yocto Project is an open source collaboration project that provides templates, tools, and methods to help you create custom Linux*-based systems for embedded products, regardless of the hardware architecture.

The de facto industry standard “tool kit” for building custom embedded Linux operating systems with ***over 50% market share by volume and over 80% by revenue***

- Provides common format/repository for Linux* Board Support Packages (BSPs) for easy porting, sorted by Architecture
- Helps manage adherence to Open Source Licensing thru Filtering and Manifest creation and archiving
- Compliance Program promotes interoperability and enables an easy transition from proof of concept (POC) to supported commercial Linux (Wind River, Monta Vista, ENEA and TimeSys)
- Generates a custom designed application development kit for each specific device
- Simple to port across architectures

Simple Electronics 

IoT 

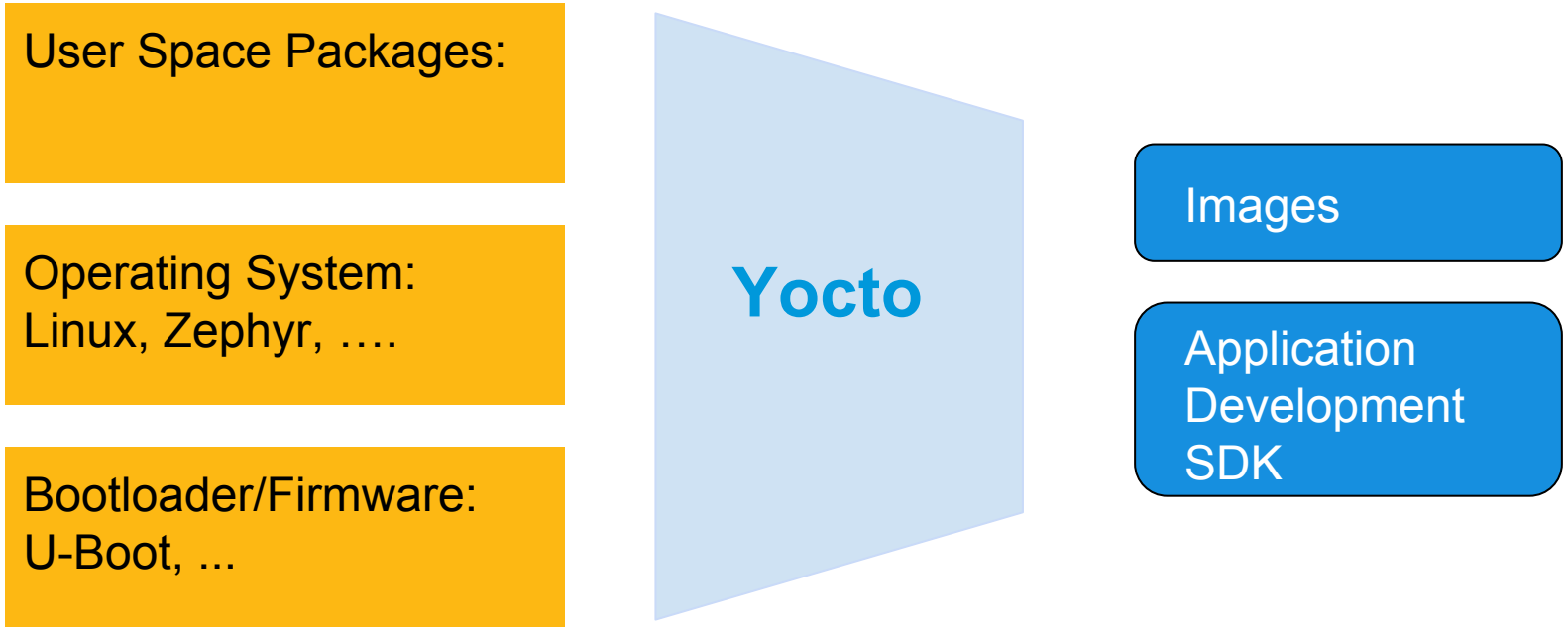
Point of Sale 

Networking & Storage 

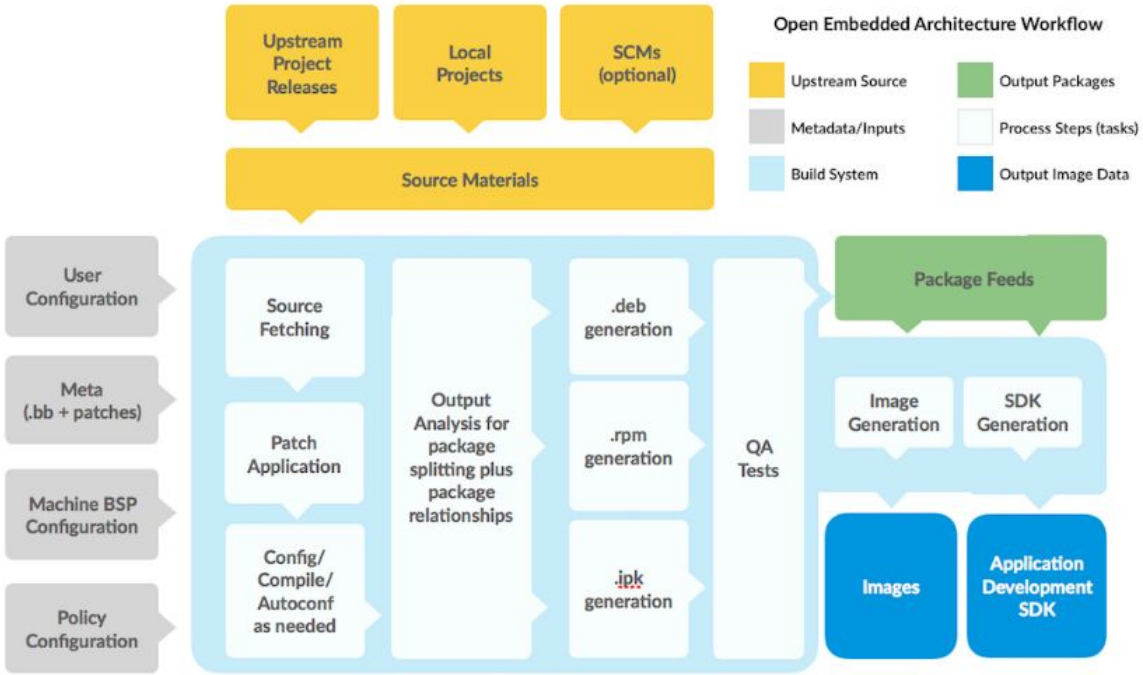
Industrial 

It’s Not an Embedded Linux Distribution, it Creates a Custom One for You
Learn More at www.yoctoproject.org

Building Embedded Systems



Building Images with Yocto & Open Embedded



Proposal: Generate SBoMs

User Space Packages:

Operating System:
Linux, Zephyr, ….

Bootloader/Firmware:
U-Boot, ...

Images + SBoMs

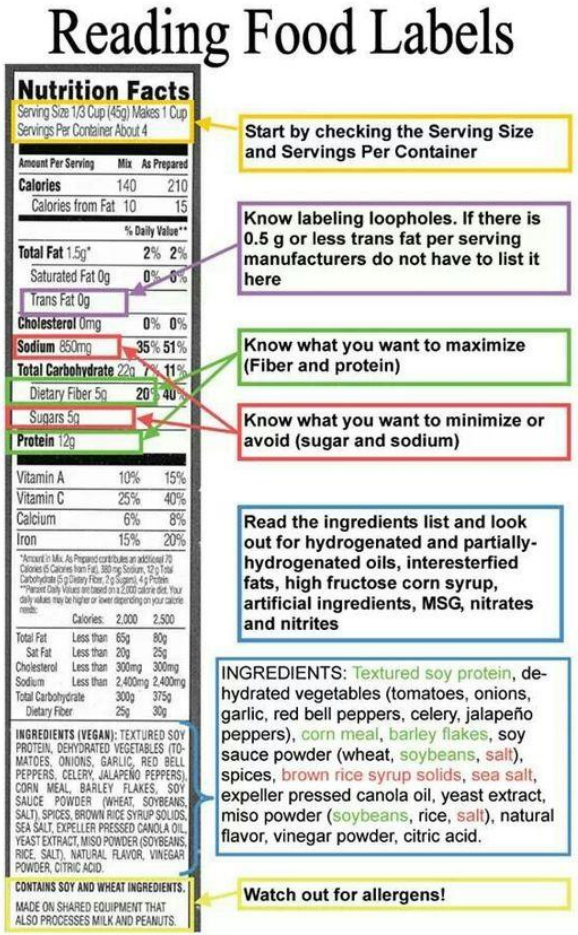
Application Development SDK + SBoM

What is a software bill of materials (SBoM) ?

Software Bill of Materials

A **software bill of materials** (software BOM) is a list of components in a piece of software. Software vendors often create products by assembling open source and commercial software components. The software BOM describes the components in a product.[1][2] It is analogous to a list of ingredients on food packaging

source: https://en.wikipedia.org/wiki/Software_bill_of_materials



source: https://www.pinterest.ca/pin/278660295677291647/

A product's components can be created from open source and proprietary software packages.

Modern open source projects are an interwoven set of multiple dependencies on other projects with multiple versions of source and licenses applying.

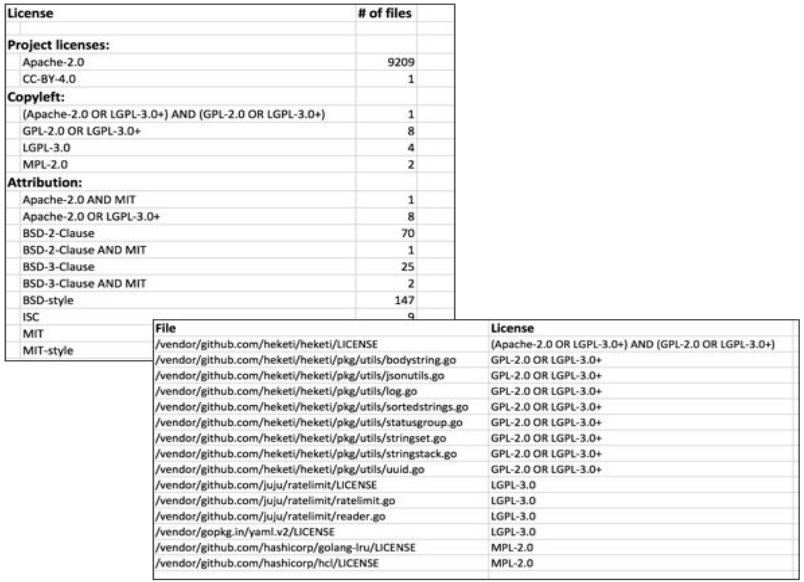
Challenge: Accurately summarizing the information

Different programming languages impose different standard ways of handling third party dependencies.

- Java (e.g. ONAP): retrieved at build time
- Go (e.g. Kubernetes, Hyperledger Fabric): checked into source code repository

Adds complexity when thinking about what components are "in" the project code.

Sharing SBOMs: What Information is Significant?



License	# of files
Project licenses:	
Apache-2.0	9209
CC-BY-4.0	1
Copyleft:	
(Apache-2.0 OR LGPL-3.0+) AND (GPL-2.0 OR LGPL-3.0+)	1
GPL-2.0 OR LGPL-3.0+	8
LGPL-3.0	4
MPL-2.0	2
Attribution:	
Apache-2.0 AND MIT	1
Apache-2.0 OR LGPL-3.0+	8
BSD-2-Clause	70
BSD-2-Clause AND MIT	1
BSD-3-Clause	25
BSD-3-Clause AND MIT	2
BSD-style	147
ISC	9
MIT	
MIT-style	

File	License
/vendor/github.com/heketi/heketi/LICENSE	(Apache-2.0 OR LGPL-3.0+) AND (GPL-2.0 OR LGPL-3.0+)
/vendor/github.com/heketi/heketi/pkg/utils/bodystring.go	GPL-2.0 OR LGPL-3.0+
/vendor/github.com/heketi/heketi/pkg/utils/jsonutils.go	GPL-2.0 OR LGPL-3.0+
/vendor/github.com/heketi/heketi/pkg/utils/log.go	GPL-2.0 OR LGPL-3.0+
/vendor/github.com/heketi/heketi/pkg/utils/sortedstrings.go	GPL-2.0 OR LGPL-3.0+
/vendor/github.com/heketi/heketi/pkg/utils/statusgroup.go	GPL-2.0 OR LGPL-3.0+
/vendor/github.com/heketi/heketi/pkg/utils/stringset.go	GPL-2.0 OR LGPL-3.0+
/vendor/github.com/heketi/heketi/pkg/utils/stringstack.go	GPL-2.0 OR LGPL-3.0+
/vendor/github.com/heketi/heketi/pkg/utils/uuid.go	GPL-2.0 OR LGPL-3.0+
/vendor/github.com/juju/ratelimit/LICENSE	LGPL-3.0
/vendor/github.com/juju/ratelimit/ratelimit.go	LGPL-3.0
/vendor/github.com/juju/ratelimit/reader.go	LGPL-3.0
/vendor/gopkg.in/yaml.v2/LICENSE	LGPL-3.0
/vendor/github.com/hashicorp/golang-lru/LICENSE	MPL-2.0
/vendor/github.com/hashicorp/hcl/LICENSE	MPL-2.0

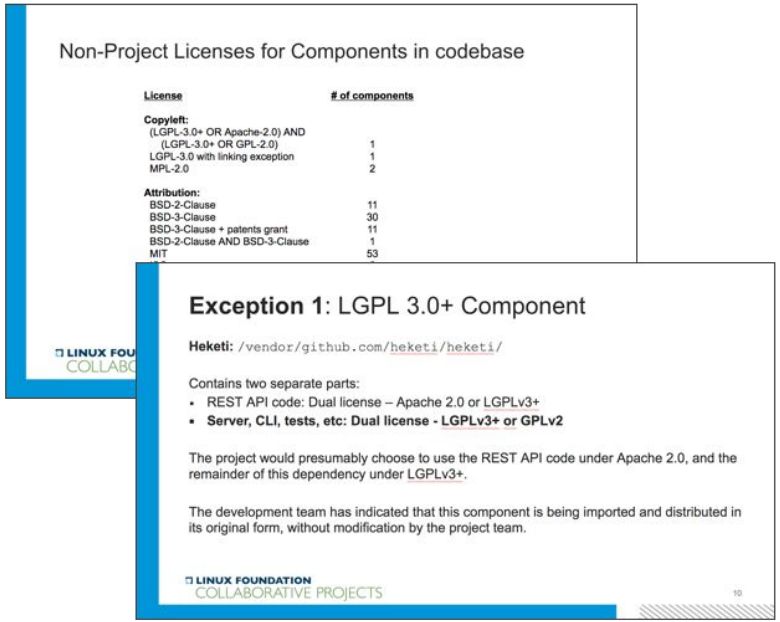
Information for Developers

List of packages being used.

Includes name, version number, checksums, download location, source location, license information, build and run dependencies, vulnerability identifiers etc.

Information permits tooling to build up a product and track components

Sharing SBOMs: What Information is Significant?



Non-Project Licenses for Components in codebase

License	# of components
Copyleft:	
(LGPL-3.0+ OR Apache-2.0) AND (LGPL-3.0+ OR GPL-2.0)	1
LGPL-3.0 with linking exception	1
MPL-2.0	2
Attribution:	
BSD-2-Clause	11
BSD-3-Clause	30
BSD-3-Clause + patents grant	11
BSD-2-Clause AND BSD-3-Clause	1
MIT	53

Exception 1: LGPL 3.0+ Component

Heketi: /vendor/github.com/heketi/heketi/

Contains two separate parts:

- REST API code: Dual license – Apache 2.0 or LGPLv3+
- **Server, CLI, tests, etc: Dual license - LGPLv3+ or GPLv2**

The project would presumably choose to use the REST API code under Apache 2.0, and the remainder of this dependency under LGPLv3+.

The development team has indicated that this component is being imported and distributed in its original form, without modification by the project team.

LINUX FOUNDATION
COLLABORATIVE PROJECTS

10

Information for Management

Summary of license findings and security information, tailored for discussion with legal counsel and executives for risk management.

May include findings, recommendations and information to assist with evaluating exception requests

Software “ingredient” management has not been standardized:

- Varies by language, distro, repository, company...
- Incomplete or inconsistent license notices and security info
- Time-consuming to manually examine files
- Time-consuming to maintain scanning processes

Most developers want to build software, not spend time dealing with packaging, labeling and distribution.

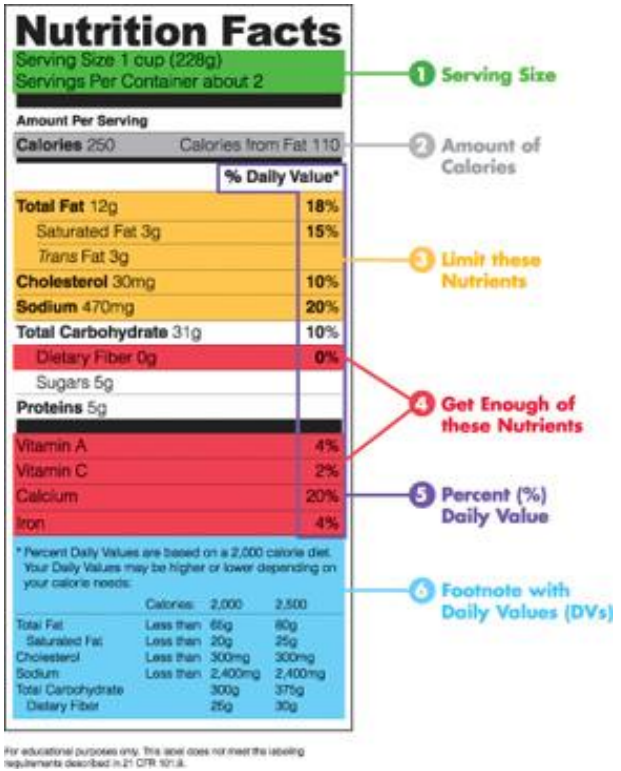
What's Done Today?

- Run scans on incoming project codebases and patches
- Analyze license texts, notices and security references to determine if can use internally, compose with other components, distribute
- Produce ad-hoc summary reports of **"key information"** for builds, releases, patches, archives, etc. to share with consumers
- Try to apply **"best practices"** as understood within projects, communities, organizations and companies

“Key Information”

- Varies from project to project
- Varies from tool to tool
- Varies from company to company
- Varies from community to community

How do we agree on these“Nutrition Facts” we want to keep with our “Ingredients”?



BOM Documentation (1)

BOM: „Bill of Material"

- It is a general question what is in the delivery
- Understand the nature of the delivery (How much OSS?)
- Understand potential issues (IP)
- How else to ensure license compliance?
- Basics of supply chain issues actually apply also to software
- Software Package Data Exchange (SPDX) specifies one implementation how to express a BOM of a software package [1]

[1] https://spdx.org/

BOM Documentation (2)

Bill of material can be general obligation, for example at:

- USA: Cyber Supply Chain Management and Transparency Act of 2014
- Germany: KRITIS: BSI-Kritisverordnung [2]
  - Obliged to report service disturbances
  - Obliged to implement information security
  - Requires knowledge about BOM

[2] https://www.bmi.bund.de/SharedDocs/pressemitteilungen/DE/2017/06/nis-richtlinie.html

But we do have a Language to “Exchange” Facts

Software Package Data Exchange® (SPDX®)

is an open standard for communicating software bill of material information (including components, licenses, copyrights, and security references).

Openly Created SBoM Format for the “Nutrition Facts”

The Software Package Data Exchange (SPDX®) Specification Version 2.1.1

With thanks to Adam Cohn, Andrew Back, Ann Thornton, Bill Schineller, Bruno Cornec, Ciaran Farrell, Daniel German, David Wheeler, Debra McGlade, Dennis Clark, Ed Warnicke, Eran Strod, Eric Thomas, Esteban Rockett, Gary O'Neall, Guillaume Rousseau, Hassib Khanafer, Jack Manbeck, Jaime Garcia, Jeff Luszcz, Jilayne Lovejoy, John Ellis, Karen Copenhaver, Kate Stewart, Kevin Mitchell, Kim Weins, Kirsten Newcomer, Kris Reeves, Liang Cao, Marc-Etienne Vargenau, Mark Gisi, Marshall Clow, Martin Michlmayr, Martin von Willebrand, Matt Germonprez, Michael J. Herzog, Michel Ruffin, Nuno Brito, Oliver Fendt, Paul Madick, Peter Williams, Phil Robb, Philip Odence, Philip Koltun, Phillippe Ombredanne, Pierre Lapointe, Rana Rahal, Robin Gandhi, Sam Ellis, Sameer Ahmed, Scott K Peterson, Scott Lamons, Scott Sterling, Shane Coughlan, Steve Cropper, Stuart Hughes, Tom Callaway, Tom Vidal, Thomas F. Incorvia, Thomas Steenbergen, Venkata Krishna, W. Trevor King, Yev Bronshteyn, and Zachary McFarland for their contributions and assistance.

https://spdx.github.io/spdx-spec/

Used to communicate software identification, license and security information in standardized, machine-readable formats

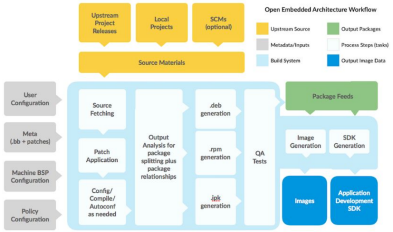
SPDX files can be produced from source code scans or **builds**, curated and annotated by reviewers, and shared between organizations

Based on **8 years of analysis of use cases**, incorporating input from industry experts in packaging, licensing and security

Building on the Yocto Builds...

THE LINUX FOUNDATION

Overview of the Yocto Project Builds



Fetches and cross-compiles package (e.g. linux) from source code

Minimal host dependencies (see HOSTTOOLS in bitbake.conf, coreutils, git, python and small number of others)

'Recipes' specify license of software, e.g. `LICENSE = "MIT"`

License files are specified and checksummed - if they change, LICENSE may change (often just copyright year or address of FSF):

```
LIC_FILES_CHKSUM = "file://COPYING;md5=f27defe1e96c2e1ecd4e0c9be8967949 \
file://sed/sed.h;beginline=1;endline=17;md5=767ab3a06d7584f6fd0469abaec4412f
"
```

Support in Yocto for License Management

Can map ‘yocto’ license values to SPDX license list

- SPDXLICENSEMAP[GPLv3] = "GPL-3.0"
- See: http://git.yoctoproject.org/cgit.cgi/poky/tree/meta/conf/licenses.conf

Have standardised SPDX license list sources available:

- http://git.yoctoproject.org/cgit.cgi/poky/tree/meta/files/common-licenses

Can filter builds to exclude specific licences

Generates list of licenses present in given image, license texts and awareness of some implications (e.g. sharing source code)

Basically any processing can be implemented/customised

Example: Analysis of Linux Stack Source

Create sourcestats.bbclass:

```
SOURCESTATS_FILE = "${TMPDIR}/sourcestats.txt"
SOURCESTATS_FILE_class-native = "${TMPDIR}/sourcestats-native.txt"

do_sourcestats () {
    cd ${S}
    files=`find -type f | wc -l`
    spdx=`grep SPDX-License-Identifier: -r -I --exclude-dir=temp . 2> /dev/null | wc -l`
    size=`du -sb . | cut -f 1`
    echo "${PN},$files,$size,$spdx" >> ${SOURCESTATS_FILE}
}
addtask sourcestats after do_patch before do_prepare_recipe_sysroot
```

In local.conf: `INHERIT += “sourcestats”`

Then “`bitbake core-image-sato`”

Results for "core-image-sato"

core-image-sato: busybox Linux image with an X11 gtk+ desktop

Files with SDPX header:

36,540

Files Total:

579,668

```
linux-libc-headers: 30% (18678/61718)  (4.19 kernel)
linux-yocto: 28% (17608/61509)  (4.18 kernel)
alsa-utils: 12% (34/282)
libgpg-error: 9% (28/282)
dtc-native: 5% (17/314)
kern-tools-native: 4% (9/210)
libevdev: 1% (3/187)
btrfs-tools: 0% (5/568)
libgcrypt: 0% (3/517)
qemu-native: 0% (76/16540)
libdrm: 0% (1/372)
libinput: 0% (2/327)
```

Yocto Project binary “debug” info

Used for debugging/profiling

- Builds include debug info by default (`CFLAGS += -g`)
- Debug info split into separate linked files and then packaged separately (sed and sed-dbg packages)
- Impacts build time but not final image size
- Already include source code in XXX-src packages based on debug info

but missing an opportunity - licensing?

Debug Information: What's There?

dwarfsrcfiles - Simple single file C program using elfutils

http://git.yoctoproject.org/cgit.cgi/poky/tree/meta/recipes-devtools/dwarfsrcfiles/files/dwarfsrcfiles.c

```
$ dwarfsrcfiles libXrender.so
/usr/src/debug/glibc/2.29-r0/git/csu/../sysdeps/x86_64/crti.S
/usr/src/debug/glibc/2.29-r0/git/csu/../sysdeps/x86_64/crti.S
/usr/src/debug/libxrender/1_0.9.10-r0/build/src/../../libXrender-0.9.10/src/AddTrap.c
/usr/src/debug/libxrender/1_0.9.10-r0/build/src/../../libXrender-0.9.10/src/AddTrap.c
/usr/include/bits/string_fortified.h
/usr/lib/x86_64-poky-linux/gcc/x86_64-poky-linux/8.3.0/include/stddef.h
/usr/include/X11/X.h
/usr/include/X11/Xlib.h
/usr/include/X11/Xlibint.h
/usr/include/X11/Xmd.h
/usr/include/X11/Xproto.h
/usr/include/X11/extensions/render.h
/usr/include/X11/extensions/renderproto.h
/usr/src/debug/libxrender/1_0.9.10-r0/build/src/../../libXrender-0.9.10/include/X11/extensions/Xrender.h
/usr/src/debug/libxrender/1_0.9.10-r0/build/src/../../libXrender-0.9.10/src/Xrenderint.h
/usr/src/debug/libxrender/1_0.9.10-r0/build/src/<built-in>
```

Combining Capabilities Together?

List of source files that make up any executable

+

SDPX headers of source files

⇒ License???

How easy to add custom processing?

http://git.yoctoproject.org/cgit.cgi/poky-contrib/log/?h=rpurdie/license-experiments-osls

Two commits:

package: Generate srclist and filelics json files

Change do_package to:
a) Save the debug object -> source file mapping information (in TEMPDBGSRCMAPPING)
b) Save a .srclist file containing that information (in json format)
c) Perform a grep of all the sources in the build of this recipe looking for SPDX license headers
d) Save out a file listing the SPDX headers found for each file

package: Add license computation experiment

This code creates a list of SPDX headers found for the sources that make up a given set of binaries that make up an individual package.

This is then compared with the license field of the given package containing those binaries.

Combining Capabilities Illustrate Concerns

Image	Extracted from debug file	Recipe LICENSE
libgcrypt-1.8.4-r0	LGPL-2.1+	LGPLv2.1+
glibc-2.29-r0	GPL-2.0 WITH Linux-syscall-note	GPLv2 & LGPLv2.1
busybox-1.30.1-r0	GPL-1.0+ WITH Linux-syscall-note, GPL-2.0 WITH Linux-syscall-note	GPLv2 & bzip2
coreutils-8.30-r0	GPL-2.0 WITH Linux-syscall-note	GPLv3+
wpa-supplicant-2.7-r0	GPL-2.0 WITH Linux-syscall-note, LGPL-2.1+	BSD
libgpg-error-1.35-r0	LGPL-2.1+, LGPL-2.1-or-later	GPLv2+ & LGPLv2.1+
libpciaccess-0.14-r0	LGPL-2.0+ WITH Linux-syscall-note, GPL-2.0 WITH Linux-syscall-note	MIT & MIT-style

Visualizing Licensing Interactions

Limitation: Some licenses simply totally missing today, no SPDX headers

How licenses combine?:

- `GPL-1.0+ WITH Linux-syscall-note + GPL-2.0 WITH Linux-syscall-note = GPL-2.0 WITH Linux-syscall-note`
- `GPL-2.0 WITH Linux-syscall-note + X = X`

Interpretation is key, may need more context:

- Using a LGPL-2.1 header in software using it as a dynamically linked library → no impact on recipe license.
- Using LPGL-2.1 source code without dynamic linking would impact license compatibility

Mapping Debug Information in Yocto to SPDX documents

- Current limited coverage already raises questions about license compatibility
- SPDX spec has “Concluded Licence” but need tools + mapping data to compute
- SPDX also has copyright holders fields and other info, scan source with tools to generate?
- Engineers really interested in “Concluded obligations” for any given SBoM, not just license but any other obligations

Next Steps:

- Encourage key projects to add SPDX license header to source files
- Highlight key successes (GnuPG, u-boot) and key gaps (GLIBC)
- Standard library of license interactions?
- Improved tooling to visualise state of the source
- Collaborate to reuse existing modules/tools (fossology components?)
- Generate SBoMs from Yocto Project automatically

Yocto Goal: Each deployed artefact has SPDX format SBoM?


Thank You !
Any questions?
strategy & planning
analysis
develop
execute
idea
plan
training
innovate
process
materials
roll out
search
business plan
trend analysis
marketing plan
development strategy
long term goals
team building
cloud
business
risk
goal
target
strategy
return
risk
time
money
personal resources
35%
26%
27%
10%
25%
50%

What is an SPDX Document?

tag:value

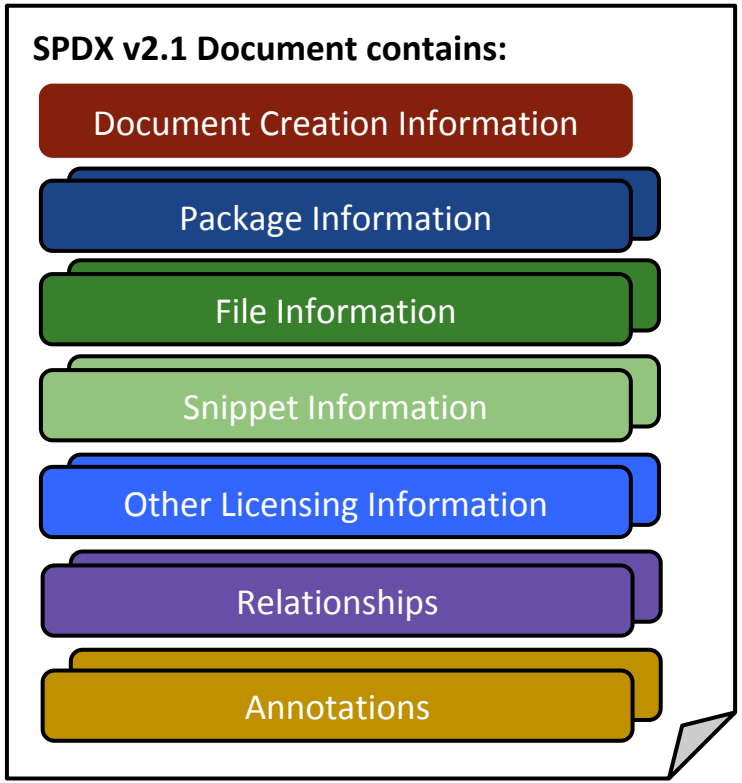
```
##-------------------------
## Package Information
##-------------------------

PackageName: time-1.7.tar.gz
PackageFileName: time-1.7.tar.gz
PackageDownloadLocation: NOASSERTION
PackageVerificationCode: dd5cf0b17bfef4284c6c22471b277de7beac407c
PackageChecksum: SHA1: dde0c28c7426960736933f3e763320680356cc6a
PackageLicenseConcluded: GPL-2.0+
PackageLicenseDeclared: GPL-2.0+
PackageLicenseInfoFromFiles: GPL-2.0
PackageLicenseInfoFromFiles: GPL-2.0+
PackageLicenseInfoFromFiles: MIT
PackageLicenseInfoFromFiles: LicenseRef-1
PackageLicenseInfoFromFiles: LicenseRef-2
PackageLicenseInfoFromFiles: LicenseRef-3
PackageCopyrightText: NOASSERTION

##-------------------------
```

RDF/XML

```
dolph Chung &lt;tausq@debian.org&gt;

://spdx.org/licenses/GPL-2.0+"/>
p://spdx.org/rdf/terms#noassertion"/>
g/rdf/terms#fileType_source"/>

eName>
g/rdf/terms#File"/>

- <rdf:Description rdf:nodeID="A2">
    <checksumValue>dc90a437e03f31ab04e7059d8da5f88b28cde77d</checksumValue>
    <algorithm rdf:resource="http://spdx.org/rdf/terms#checksumAlgorithm_sha1"/>
    <rdf:type rdf:resource="http://spdx.org/rdf/terms#Checksum"/>
  </rdf:Description>
- <rdf:Description rdf:nodeID="A3">
    <rdfs:comment/>
    <copyrightText rdf:resource="http://spdx.org/rdf/terms#none"/>
    <licenseComments/>
```

SPDX

What makes up an SPDX Document?



SPDX

Only subset of fields are mandatory

Document Creation Information

1 per document

- 2.1 SPDX Version.
- 2.2 Data License
- 2.3 SPDX Identifier
- 2.4 Document Name
- 2.5 SPDX Document Namespace
- 2.8 Creator
- 2.9 Created

1 per package in document

Package Information

- 3.1 Package Name
- 3.2 Package SPDX Identifier
- 3.7 Package Download Location
- 3.9 Package Verification Code
- 3.13 Concluded License
- 3.14 All Licenses Information from Files
- 3.15 Declared License
- 3.17 Copyright Text

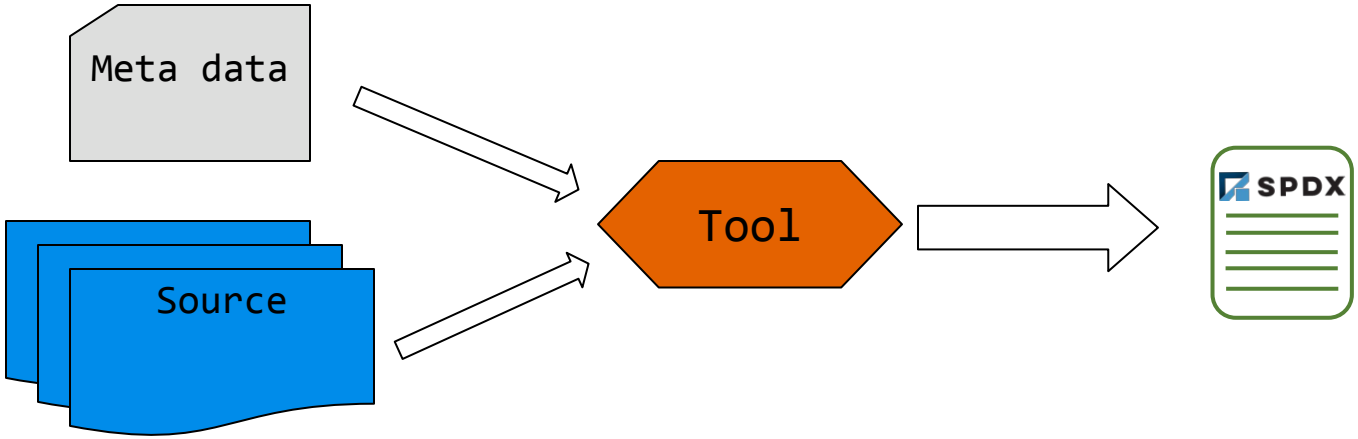
File Information

- 4.1 File Name
- 4.2 File SPDX Identifier
- 4.4 File Checksum
- 4.5 Concluded License
- 4.6 License Information in File
- 4.8 Copyright Text

1 per file in each package

SPDX

Creating an SPDX document



Requires tooling as the package verification code is generated by file **checksums** and the total **number of files** per useful package

SPDX