THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
NORTH AMERICA

Embedded Linux Conference

SPDX and SBOM work for the Linux kernel

Tim Bird
Principal Software Engineer, Sony

Abstract

Due to increased interest in fine-grained analysis of kernel composition and security (due to the CRA and other recent cybersecurity legislation), there have been a number of recent projects to 1) generate SBOMS for the linux kernel, and 2) finish adding the remaining SPDX-License-Identifier lines to the kernel source tree. In this talk, Tim will describe the current status of both of these efforts. Good progress has been made to add missing SPDX id lines, but more work is needed to complete this project. Tim proposes a kselftest test to make sure there are no regressions in this area. The status of different kernel SBOM generation tools, and upstream status, will be described.

This work should be of interest to companies interested in complying with cybersecurity requirements, as well as those involved with license compliance efforts.

Outline

- SBOM work in the kernel
- SPDX work in the kernel
- Contributing SPDX lines to the kernel
- You can contribute also!

SBOM work in the Linux kernel

THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
NORTH AMERICA
Embedded Linux Conference

SBOM systems for the kernel

- KernelSbom
- ESSTRA
- Others

KernelSBom

- Overview
- How it works
- What it misses/what could be improved

KernelSBom overview

- KernelSBom patch set (and github project)
  - Original submission V1 on February 10, 2026
    - First heard about this in December 2025, at Linux Plumbers Conference
- By Luis Augenstein and Maximillian Huber
- Looks like a strong candidate for acceptance upstream
  - New (V6) patch set contributed on May 7
  - https://lore.kernel.org/linux-kbuild/20260507173827.70949-1-luis.augenstein@tngtech.com/
- Documentation in Documentation/tools/sbom/sbom.rst (when accepted)
  - or for an early preview, here: https://lore.kernel.org/linux-kbuild/20260507173827.70949-2-luis.augenstein@tngtech.com/
- Kernel commit tags say:
  - Assisted-by: Cursor:claude-sonnet-4-5
  - Assisted-by: OpenCode:GLM-4-7
  - That's interesting (IMHO this is the kind of thing AI can be trusted with)

How it works

- KernelSBom uses .cmd files (from kernel build system) to create a dependency graph
  - Kernel build system ('Kbuild') creates '.cmd' files to record dependencies
    - ex: in $KBUILD_OUTPUT/init -> .main.o.cmd (for init/main.c)
    - .cmd file is in Makefile format - with target dependencies
      - 'savedcmd' variable has the exact command used to build the program
  - Kbuild uses fixdep and macros defined in scripts/Kbuild.include and scripts/Makefile.build
  - Kbuild uses dependency file (.d) generated by the compiler
- KernelSBom parses .cmd files after a build, and produces json files:
  - sbom-build.spdx.json, sbom-output.spdx.json, sbom-source.spdx.json
  - Files are in SPDX version 3.0.1 format.
- Some tools used during build don't generate any dependency files, and thus don't have .cmd files
  - Have to handle those individually

What it misses

- Testing indicates it gets 99% of of build dependencies
  - Is the most comprehensive build tracker for the Linux kernel that I've seen
- Can miss scripts or shell fragments (see next slide)
- Allows for explicit dependencies, if needed
  - scripts/sbom/sbom/cmd_graph/hardcoded_dependencies.py
  - It is too difficult to parse Makefiles and shell script fragments
- There's an option to continue even if errors are encountered
  - So, can generate a possibly incomplete source SBOM

Problem I encountered

- Got a problem on arm64 build
  - parsing command: gen-kernel-hwcaps.sh build script
- Worked around it by running tool manually with "–do-not-fail-on-unknown-build-command"
  - Ran make with V=1, and copy sbom command line, adding SRCARCH=arm64

```
 GEN     sbom-source.spdx.json sbom-build.spdx.json sbom-output.spdx.json
[ERROR] File
"/home/tbird/work/torvalds/linux/scripts/sbom/sbom/cmd_graph/savedcmd_parser/savedcmd_parser.py", line 33, in log_error_or_warning
Skipped parsing command /bin/sh -e
/home/tbird/work/torvalds/linux/arch/arm64/tools/gen-kernel-hwcaps.sh
/home/tbird/work/torvalds/linux/arch/arm64/include/uapi/asm/hwcap.h >
arch/arm64/include/generated/asm/kernel-hwcap.h because no matching parser was found
```

KernelSBom Resources

- Materials from LPC 2025 talk:
  - Slides:
    - https://lpc.events/event/19/contributions/2306/attachments/1836/3962/KernelSBOM-Maximilian-Huber.clicks%20(1).pdf
  - Video: https://youtu.be/rTHMeiI7Tto
- Github repository: https://github.com/TNG/KernelSbom

ESSTRA overview

- Sony's tool for **E**nhancing **S**oftware **S**upply Chain **Tra**nsparency
- Uses a gcc plugin
- Embeds source information into an ELF section in the binaries
  - Binaries carry the source/build data with them
  - Linker continues flowing data through build
  - Great for getting information from flaky suppliers!!
- There is no effect on the binary behavior
- Is a general tool
  - ESSTRA developers have been working with Debian dev for a full SBOM-able Linux distro
  - I used a special version with hooks for the kernel build system

ESSTRA additional notes

- It misses things not built with compilers
  - ESSTRA has prototype LLVM and go support
  - But still have other code generation or transformation tools in kernel build
- Can be transparent to the build system
  - Can install a GCC Spec file that invokes the plugins for the compiler and linker automatically
- Can use extra tools to extract license information and generate SBOMs
- There's a size overhead for embedding the source file info into binaries

ESSTRA Resources

- Materials from Open Source Summit EU talk:
  - Slides: https://birdcloud.org/bc-files/20250925_OSSEU2025_ESSTRA.pdf
  - Video: https://youtu.be/xCYLJpCDlOE
- Github repo: https://github.com/sony/esstra

Others...

- Yocto Project – has sandbox around build and can monitor actions
  - Collects information during build and can generate SBOMS by tweaking project config
  - https://docs.yoctoproject.org/next/dev-manual/sbom.html
  - Is specific to embedded
- Heimdall – post-build scanner, with toolchain plugins
  - Is not kernel-specific
    - Only handles binaries
  - https://github.com/fossabot/heimdall-3

Resources:

- Improving Build Time SBoMs – talk by Joshua Watt at LPC 2025
  - Slides: https://lpc.events/event/19/contributions/2020/attachments/1767/3994/Improving%20Build%20Time%20SBoMs-3.pdf
  - Video: https://youtu.be/VlrS08U5g3U

SPDX work in the Linux kernel

THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
NORTH AMERICA
Embedded Linux Conference

SPDX work

- Rationale for SPDX identifier lines
- Kernel history
- Previous SPDX identifier efforts
- Work scope and priorities
- Status as of May 2026
- Issues and Difficulties

Rationale for SPDX identifier lines

- Project summary: Add SPDX license identifiers that are missing in the kernel source
- Why?
  - Convert ambiguous text into well-known and accepted strings
    - Resolves license
    - Reduces license uncertainty
    - Gives original authors input into license clarification
  - Provide machine-readable data
  - Avoids scanning and license matching (which can be ambiguous)
  - Allows for more definitive SBOM creation

Kernel History

- Linux kernel development
  - How Linus worked in the early days (patch aggregator)
  - How it works today – patches managed in git
- Linux kernel licensing history
  - Important dates in kernel licensing history:
    - Kernel was always GPL 2.0 (from inception - Sept. 1991)
    - COPYING file was formally added Sept. 1992
    - system call exception added in July 1993
    - "only valid [GPL] version [is 2]" notice added Sept. 2000
      - codified the "no 'or-later'" policy
    - DCO was added in June 2004
    - formal statement saying (no GPL 3.0) in Sept. 2006

Kernel Licensing confusion

- Kernel allows individual files to have separate and dual licensing
  - Many other OSS projects require a single license
- Linux development community wanted to be fair to original authors
  - Started with BSD code
  - Could have converted to GPL-2.0, but wanted to allow code (ie individual drivers) to be shared back with original BSD authors and projects
- And so, there are lots of files with weird headers
  - In the early days (1990s and early 2000s), OSS developers created or customized licenses
  - Ugh!
- There are corner cases where the license on an individual file is unclear
  - NOTE: Everything submitted to kernel must be GPL-2.0 compatible – this is not a legal problem
- End result is: **Conversion to SPDX requires human analysis**

Overall themes

- Make all license identification machine-readable
- End redundant license text scanning and human interpretation

Overall themes

- Make all license identification machine-readable
- End redundant license text scanning and human interpretation
- End license confusion

Overall themes

- Make all license identification machine-readable
- End redundant license text scanning and human interpretation
- End license confusion

Give Fossology nothing to do for the Linux kernel.

Previous SPDX identifier efforts

- SPDX identifiers were first created in 2010
- First full 1.0 SPDX spec (1.0) was released in 2011
- SDPX identifiers were added to kernel in big pushes in 2017 and 2019
  - Greg Kroah-Hartman added 8772 in 2017
  - Thomas Gleixner added 15886 in 2019
  - Most other years have been organic growth due to required inclusion
    - scripts/checkpatch.pl reports an error if SPDX id is missing
- Very little work on existing files, since 2020

Year	# of SPDX id lines introduced
2016	1
2017	9357
2018	5099
2019	20415
2020	4399
2021	3076
2022	3988
2023	3690
2024	3128
2025	3965
2026	1047 (so far)
total	58165

Work scope

- **16669** files are missing SPDX id lines in 7.1-rc2 (!!)
- But, they are in different categories (some we don't care about):
  - Runtime kernel, and the loader/decompression code
  - Stuff used for builds (source, instructions, scripts)
    - C/assembly source files
    - Build files (Kconfig, Makefiles) – license not interesting for compliance
  - Other stuff:
    - Tools (some used for build, some are miscellaneous utilities and debug tools)
    - Test programs and test data
    - Crypto keys, Firmware, Rust – some of these matter
- About 9000 source files

Priorities – which files to work on first

- (1) Sony build source files
  - Source files included in a build of the kernel with a Sony product config
    - This omits documentation, test scripts, build process files, etc.
- (2) defconfig build source files
  - Source files included in a defconfig build for arm64 or x86_64
- (3) Remaining source files
- (4) Build process files
  - Makefiles, Kconfig, build scripts, build tools source
  - First for Sony builds
  - Then for arm64 and x86_64 deconfigs
- (5) Remaining other files
  - Documentation, test scripts, samples, etc. (very low priority)
  - These don't show up in SBOMS

arm64 defconfig top directories missing counts

- **1692** files that are source for an arm64 defconfig build are missing SPDX-License-Identifier lines
- That's not too bad
  - Way less than 16K

Directory	Files missing SPDX ids
sound	1
crypto	2
arch	15
net	28
fs	33
lib	46
include	314
drivers	1252
TOTAL	1692

x86_64 defconfig top directories missing counts

- **526** files that are source for an x86_64 defconfig build are missing SPDX-License-Identifier lines
- This is manageable

Directory	Files missing SPDX ids
net	5
fs	26
arch	55
lib	37
drivers	185
include	218
TOTAL	526

Sony config* top directories missing counts

- **438** files that are source for a Sony configured* build are missing SPDX-License-Identifier lines
- *… I should be done by lunch*

Directory	Files missing SPDX ids
sound	1
crypto	2
arch	9
net	42
lib	43
fs	67
drivers	68
include	206
TOTAL	438

* Sony config is estimated from Bravia TV config for 5.x kernel, forward-ported to 7.1-rc2

Status as of May 2026

- I gave a status update on the mailing lists in February
  - https://lore.kernel.org/linux-spdx/MW5PR13MB56326322F7DC30A5A78C3D6AFD66A@MW5PR13MB5632.namprd13.prod.outlook.com/
- Have worked through: kernel, io_uring, sound, mm, ipc, and block dirs
  - Have done some files in 'net' directory so far
- Currently working on net/llc, net/Bluetooth, net/tipc
  - Have patches for these, but have not submitted them yet
- Stuff that is 'stuck':
  - sound/usb/midi.c – outlier license (see next page)
  - Some files under 'net' – Kerberos authentication with "Michigan" license
  - I maintain a list of files with problems at:
    - https://birdcloud.org/bc/Missing_SPDX_NOTES

Issues

- Developers acting as their own lawyer
  - Vague or ambiguous license statements
- Unversioned GPL references
- Maintainers no longer available, or don't have contact info
- Disagreement over what license to apply
  - Only one so far: Rusty vs. [Thomas]? in kernel/cpu.c
    - Rusty only specified "GPL" (unqualified)
    - Should it be 'GPL-2.0-or-later' vs 'GPL-2.0-only'
    - Rusty (original author) wanted GPL-2.0-or-later, but only wrote <4% of the existing code
      - Thomas was by far the biggest contributor
    - Settled on 'GPL-2.0-or-later'
- Non-copyrightable stuff
  - 2-line headers?, stubs?
  - register definitions?
  - crypto data?
- What to do with non-source files?

License Text Difficulties

- License text with no exact matching SPDX IDs
- Michigan license (Like BSD, but a bit different)
  - Handling specific named no-endorse entity
    - Michigan University, OpenVision, others
  - Used in net/sunrpc/auth_gss/gss_krb5_*.c
- HPND-sell-variant ID
  - Needs more legal analysis (and addition under LICENSES)
  - Used in some DRM files (drivers/gpu/drm/*.c)
- BSD-Source-Code[-Author?]
  - Have submitted request for new license or clarification, to SPDX community
  - See https://github.com/spdx/license-list-XML/issues/2939
  - Applies to sound/usb/midi.c, but also others
- Other License text with weird terms
  - e.g. "this notice must be retained in full"
  - Used in net/can/*.c

Test automation for missing SPDX id lines

- Very easy to check status of lines:
  - grep –L SPDX-License-Identifier: <somedir>/*
- I have a test that automatically populates a wiki page:
  - See https://birdcloud.org/bc/KTest_Missing_SPDX

```
Here is a summary of the top-level Linux kernel directories
and how many source files are missing SPDX headers
=================================================
KERNEL_SRC=/home/tbird/work/torvalds/linux
KERNEL_VERSION=v7.1-rc2-22-gd19761bc712d
TEST_DATE=2026-05-06_14:41:40
=================================================
Directory '    arch'- Files without SPDX:        2141  11%
Directory '   block'- Files without SPDX:           0   0%
Directory '   certs'- Files without SPDX:           3  25%
Directory '  crypto'- Files without SPDX:          16   8%
Directory ' drivers'- Files without SPDX:        5830  15%
Directory '      fs'- Files without SPDX:         186   7%
Directory ' include'- Files without SPDX:         503   7%
Directory '    init'- Files without SPDX:           1   5%
Directory 'io_uring'- Files without SPDX:           0   0%
Directory '     ipc'- Files without SPDX:           0   0%
Directory '  kernel'- Files without SPDX:          13   1%
Directory '     lib'- Files without SPDX:          89   9%
Directory '      mm'- Files without SPDX:           2   0%
Directory '     net'- Files without SPDX:         103   5%
Directory 'security'- Files without SPDX:          11   3%
Directory '   sound'- Files without SPDX:           6   0%
Directory '     usr'- Files without SPDX:           8  34%
Directory '    virt'- Files without SPDX:           0   0%
Total               Files without SPDX:        8912
=================================================
```

Contributing SPDX lines to the kernel

THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
NORTH AMERICA
Embedded Linux Conference

Contributing SPDX id lines to the kernel

- It's easy, here's how to do it...
- SPDX workflow
- Tools I use
- Github repository
- Issues to track

SPDX workflow

- Find files to fix
- Research file history, authorship, date introduced
- Research license statements in headers
- Find match in LICENSES/*
  - Start to cry if not there...
- In file: add SPDX-License-Identifier line, remove license text
- Create patch, test with checkpatch.pl
- Determine who to send to (create new mysend-xxx.sh script)
  - Use getmaintainer.pl, then include original authors, where possible
- Send it off...

Tools I use

- guess-license and 'grep –L SPDX-License-Identifier'
- git-when-introduced.sh, cregit, git blame
- spdx-count-ids.sh
- spvi - "**sp**dx **vi**" – wrapper around 'vi'
  - pre-fills the license ID line based on command line arg
  - Uses correct syntax for .c and .h files
- git (add, commit, format-patch)
- checkpatch.pl, get-maintainer.pl
- mysend-xxx.sh – wrapper on 'git send-email'
- See https://github.com/tbird20d/spdx-project/blob/master/tools/README.md

Github repository

- https://github.com/tbird20d/spdx-project/

These tools were useful for me (Tim Bird) in analyzing files that are missing an SPDX ID line, discovering what license to apply, applying missing license lines, upstreaming the patches created, and testing the kernel for files that are highest priority to be worked on next.

Checking file history and authorship

- https://cregit.linuxsource.org/
  - A very fancy and fast 'git blame'
  - Data only goes back to 2.6.12 – not far enough for some files
- My own historical Linux repository
  - Based on the Yoann Padioleau repository, re-hosted by Rob Landley
  - See https://landley.net/kdocs/fullhist/
  - But with additional tags for very early kernel versions (e.g. v0.0.1, v0.10)
- git-when-introduced.sh <file>
  - Shows: date introduced, commit, original author, kernel version
    - Must use history repo for early files

Misc notes

- I don't run into super-old files very often
  - Many files missing SPDX ids are from the early 2000s
  - Well after the license was clear, but before license policies for SPDX were introduced
    - scripts/checkpatch.pl first started checking for SPDX lines in v4.17-rc1 (April 2018)
- If the file has no header, it's easy: I assign 'GPL-2.0' ID
  - I may use GPL-2.0-only if other files by the same author or in the same dir are using that
- Sometimes encounter authors that are no longer active (or have changed emails)
  - Contacting original author is "best-effort", and is often not strictly needed
    - e.g. If license text is clear enough

Project home page and data

- Project Home Page
  - https://birdcloud.org/bc/SPDX_Project
- Guidelines for license analysis
  - https://birdcloud.org/bc/Guidelines_for_fixing_Missing_SPDX_lines
- Test results
  - https://birdcloud.org/bc/KTest_Missing_SPDX
  - Shows latest tests I have run manually
- Steps I used to test KernelSBom (and integrate into my SPDX work):
  - https://birdcloud.org/bc/Linux_Kernel_Missing_SPDX_ID_lines_from_build_SBOMs
  - A bit outdated, but steps still work

Conclusions

- *SBOMS are coming...*
- *SPDX work in the kernel is progressing*
- *You can help if you'd like*
  - *It's an easy task (mostly)*
  - *A great way to learn how to contribute to the kernel*
  - *Very likely to be harmless – almost no way to damage the kernel executable*
    - *All the changes should be in comments*
    - *But do a build every once in a while to make sure nothing broke*
  - *I'm happy to help you get started!!*

- *Join the effort or just watch:*
  - *Subscribe to: linux-spdx@vger.kernel.org*

Let me know your thoughts...

Thanks!

Contact me at: tim.bird@sony.com
wiki: birdcloud.org:/bc/SPDX_Project