Android GPL license compliance

Armijn Hemel

April 14, 2011

Talk overview

- GPL violations
- license violations in Android
- license violation discovery
- license violation resolution
- license violation prevention
- Q&A

I will not talk about software patents.

GPL enforcement

- is real
- is happening in Europe and US
- happens mostly for GPLv2 and LGPLv2/2.1
- is being done by both companies and individuals
- is easy to avoid

gpl-violations.org

Harald Welte founded `gpl-violations.org` in 2004 to tackle GPL license violations through education, documentation and, as a last resort, legal action.

I joined in October 2005.

`gpl-violations.org` has solved several hundreds of cases through legal action and informal action.

License violation life cycle

Mistakes happen, but they need to be resolved.

1. understanding the problem
2. acknowledging the problem
3. fixing the problem
4. preventing the problem

Violations in Android

There are two main areas where things go wrong:

- devices
- Android marketplace

I will not talk about the marketplace, but focus on devices.

Note: it is a myth that Android user space does not contain GPL code. Google ships Android userspace without GPL code, but many manufacturers add GPL code back in (like BusyBox)!

Recent cases

Two cases were very visible:

- HTC
- Android tablet incompliance

HTC

HTC assembles the software themselves, using a SDK from upstream.

HTC has consistently stepped on people's toes:

- standard response: "Please wait 60 to 90 days"
- persistent, even after being told, plus bad PR

HTC needs to fix their software release processes.

Android tablets

Kernel developer Matthew Garrett got quite angry in late 2010 and assembled a list of Android tablets that are not in compliance.

The companies on his list don't do development, but rebadge and resell, often without any clue about what software is inside their device.

Funny responses we got:

- "Our device is based on Android not on Linux"
- "You can get the source code from Google"
- "We made some changes specifically for our tablet hardware, so you can't have the source code"
- "we think the kernel code is under GLPv3 so we have 60 days to fix the issue"

The problem is not these companies, but the whole supply chain that does not implement proper software governance.

Discovering violations

Important: violating a license is not a technical issue, but a legal issue. Technical measures ("GPL compliance engineering") are only used to obtain evidence (which is often overwhelming).

Two steps need to be taken:

1. documentation analysis
2. technical analysis

Documentation analysis

The GPLv2 license needs:

1. copy of the license text
2. complete and corresponding source code for programs distributed under GPLv2, or
3. written offer for the complete and corresponding source code

Points 1 and 3 can easily be checked by non-technical people.

Note: the "legal information" tab in the Android user interface usually fullfills point 1.

Technical analysis

Technical analysis consists of two parts:

- determining the presence of GPL licensed software in binaries
- check source code if it matches said binaries

Sometimes hardware needs to be modified to get access to the juicy bits (serial port).

Analysing binaries

- extract programs from binary blobs using various techniques
- extract human readable strings from binary programs and match with source code
- look at meta information (file names, package meta data, etc.)

If you can match a significant number of specific strings it becomes statistically hard to deny reuse of GPL licensed software.

You can do this by hand using many standard Linux tools, or use an automated tool like the Binary Analysis Tool.

The Binary Analysis Tool

The Binary Analysis Tool (short: BAT) is a tool to automate scanning binaries (firmwares, programs, installers, etc.) to detect the presence of software in a binary.

Goals:

- demystifying compliance engineering by encoding processes in code
- creating a reproduceable process and results
- creating a common set of tools and a common language for looking at binaries
- taking away excuses for companies ("we did not know and it is difficult to find out")

BAT was made by Loohuis Consulting and OpenDawn, with support from Linux Foundation and NLnet Foundation.

The Binary Analysis Tool - facts

- Apache 2 licensed
- lightweight
- wraps around standard Unix tools
- spawned academic research (presented at Mining Software Repositories 2011 conference)
- just examines binaries, but draws no legal conclusions from findings

Android specific bits will be added soon!

Want a demo? Ask me later in the hallway.

Analysing source code

1. find out what is in the binary
2. check for every binary that contains GPL licensed code if there is matching source code, under a GPL compatible license
3. check if there are no accidental leftover binaries without sources in the source code archive itself

Source code scanning versus binary scanning

So what about using source code scanning tools to ensure the binary does not contain unwanted surprises?

Unless you build the binaries yourself using the provided source code you will never be able to tell what is in a binary using only a source code scanning solution!

Handling violation reports

Something we see going wrong a lot is the way violation reports are handled, or rather: are not handled.

This leads to:

- angry people
- bad PR
- misunderstandings
- long term damage with upstream projects (HTC was told to “wait 60 to 90 days” when they needed help on LKML)

Fixing communications

- instruct support staff to hand off requests for source code. Don't let them just come up with something.
- establish a separate point of contact regarding open source license questions, preferably a team (to avoid "single point of failure") and advertise this on your website/in documentation.
- record all cases in a request tracker or CRM system, so there is a history.

Fixing a violation

Ideally:

- make complete and corresponding source code available
- inform customers of their rights
- take preventive measures so it never happens again

If this is not possible:

- stop distribution completely
- take preventive measures so it never happens again

License violation prevention

The best way to solve a license violation is to have never have it happen in the first place!

There are two components:

- technical process improvements
- non-technical process improvements

Technical process improvements

- check releases (source code and binary) from upstream. Trust, but verify!
- provide compliant releases downstream
- integrate compliance into your development process
- actively teach your engineers about licensing and when it is OK to combine/reuse software
- push your changes to upstream projects, so you cannot accidentily forget to include them the next time (plus all the other advantages!)

Non-technical process improvements

- push compliance upstream (if you source from companies) through contracts
- demand a “software bill of materials” in SPDX (Software Package Data Exchange) or another format and verify it with a checker
- check and record every release that is distributed, including beta/test versions
- use SPDX (Software Package Data Exchange) or another format to clearly indicate what you are shipping

Contact

- `armijn@gpl-violations.org` for `gpl-violations.org` related questions
- `armijn@uulug.nl` for consultancy questions

or just talk to me in the hallway.

Q&A

Remember: no questions about software patents