GNU GPL License Compliance

With specific focus on Embedded Devices

Harald Welte

gpl-violations.org
gnumonks.org
hmw-consulting.de

COSCUP 2010

Outline

About the speaker

- Using + playing with Linux since 1994
- Kernel development since 1999
- IT security expert, focus on network protocol security
- Board-level Electrical Engineering
- System-level Software for PPC, ARM, x86
- IANAL, but companies not complying with the license forced me to spend lots of time with legal issues

Free Software
Definition by the FSF

Free Software has to ensure the following key freedoms:

- Freedom to use the software for any purpose
- Freedom to make copies "to help your neighbor"
- Freedom to study its functionality (source code)
- Freedom to fix it yourself (make modifications)

Copyleft
A concept to ensure Freedom

Copyleft is an idea to use copyright to ensure Software Freedoms

- Use/claim copyright on the software
- Create a license that is permissive enough for the 4 Freedoms
- However, put some conditions/obligations in the license
  - ensure the source code will always be available
  - ensure nobody is able to remove the 4 Freedoms from the software
- Use that license for the software.

The GNU GPL

An implementation of Copyleft

The GNU General Public License (GPL)

- is a Copyleft Free Software License
- assures the original author that his work will always have the freedoms
- establishes a level of fairness: You can use my code, if you share your additions back with us.
- is a big motivation factor for many community members

Revisiting the GPLv2 License Terms

The GNU GPLv2

- Regulates distribution, not use (running the program)
- Allows distribution of source code and modified source code, if
  - The license is mentioned
  - A copy of the license text accompanies each copy
- Allows distribution of or modified binaries, if
  - The license is mentioned
  - A copy of the license text accompanies each copy
  - The source code is either included with the copy, or a written offer is made on how the source can be obtained.

Complete Corresponding Source Code

As required by GPLv2

. . . complete source code means all the source code for all modules it (the software) contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable.

- For a C language program, this means
  - Source Code
  - Makefiles
  - compile-time configuration (e.g. kernel .config)
- General rule
  - Intent of the license is to enable the user to run modified versions of the program
  - If you provide everything needed for that, there will be no discussion

Modifications of GPL'd source code

The details that matter

- In the GPL, it does not matter if you have modified the GPL'd program or if you ship it unmodified.
- You always have to provide the source code!
- If you modify the source code, your changes have to be visible/identifiable
- For practical reasons, I suggest shipping original upstream tarball + a diff/patch with your changes

Complete + Corresponding Source

For every Release you make

- Whenever you *distribute* GPL licensed software, the license applies. This includes
  - Actual sale of a physical embedded device with the software in flash
  - Download of a firmware update as a file from a website
  - Shipping of firmware updates on physical storage
  - Distribution of firmware updates e.g. by over-the-air mechanisms in DVB-S or other networks
- Every time, the conditions of the license have to be fulfilled (mention there's software under GPL, include full license text, include or offer complete corresponding source code
- For every release you ever ship (even beta release if it ever is shipped only to one customer), you need the *complete corresponding* source code.

Derivative Works

Keeping it clean

Derivative works are a question of copyright law, not the GPL

- whenever you couple a GPL and a non-GPL program tightly (e.g. static/dynamic linking), your're entering a legal grey area
- there is little or no precedent on derivative works of software
- you're violating the intention of the author. If he wanted you to link from proprietary programs, he would have used LGPL
- try to work *with* the community, rather than against it

Intermission

Take a break, go one step back

- The License is not a means to itself
- Intent of the license is to make sure people can modify + enhance the product
- The more open your product is, the less you have to worry
- Using Linux + FOSS without enabling community to modify+enhance is cheating!
- Try to make friends of the developer community, not enemies!

License compliance is not an afterthought

Complying with the license terms is relatively easy *if* you consider the license terms *before* starting R&D

- you can integrate building source releases in your build process
- you can decide which software can be combined given the license terms

License compliance is not an afterthought

Achieving license compliance after shipping the product is very hard

- lack of good engineering practise could mean old source code is gone
- engineers working on the product might have left the company
- you and your customers are under a lot of time pressure (legal threat)
- you might have already shipped a derivative work to GPLd software and now have to release parts that you originally wanted to keep proprietary

Linux and Free Software (FOSS) everywhere

Areas of Embedded Linux

- Embedded Network Devices (DSL-Modem, Router, WiFi-AP, NAS)
- Telecommunications equipment (Switch, DSLAM, ...)
- In-flight / In-vehicle entertainment
- Personal Navigation Devices (Tomtom GO)
- Mobile Phones (EZX, MAGX, Android, LiMo, WebOS)
- PoS terminals, ATMs, Payphones
- Digital Media Players, Set-Top-Boxes, Video Recorder
- Exercycles + Fitness Gear
- Building automation + control
- VoIP telephones, VoIP switches, PBX
- e-Ink readers, Tablet computers, MIDs

Embedded Linux Supply Chain

In a typical case, the supply chain consists minimal of

- The silicon maker of the SoC containig the core that runs Linux
- The supplier of the reference design / board for that SoC
- The ODM building an actual circuit board using that SoC
- The OEM selling the product under his brand in the target market

Embedded Linux Supply Chain

Situation can be further complicated by

- A 3rd party supplier of the BSP / SDK for the SoC or reference board
- Multiple companies involved on the ODM or OEM side (building parts of a product, later integration into the real product e.g. IVE for a car)
- 3rd party suppliers of application programs (which might use FOSS)

Embedded Linux Supply Chain

Problems in the supply chain:

- OEM has no clue what kind of software ODM put into the product
- ODM has limited technical skill and has no clue what BSP provider did
- End user buys a product with license/copyright violations and has no clue
  - who the entities in the supply chain are
  - who actually caused the license/copyright violation

GPL and Embedded Systems

Interpreting the meaning

- The GNU GPLv2 was written for the GNU project, at the time this project was working on replacing individual application programs on top of a proprietary UNIX operating system kernel.
- scripts used to control compilation and installation
  - Intent: To enable the user to modify + run modified versions
  - In case of embedded systems, the "scripts used to control installation" include the software required for installing the program onto the target device

GPL and Embedded DRM

Sometimes called Tivo-ization

- Some companies want to lock down their Linux-based system, by
  - Cryptographic verification of bootloader by ROM loader
  - Cryptographic verification of kernel image by bootloader...
- This is problematic from a GPL point of view, since
  - You are depriving the user from practically exercising his right to run modified versions of the program
  - Thus, violation not of the GPLv2 wording, but likely of the GPL's intention
  - Legal outcome unclear, different scholars have different opinions, also depends on jurisdiction
- GPLv3 makes this intent explicit in the license text

GPL Violations

- GPL violations are not new, just like GPL licensed software is not new
- However, increased popularity of GNU/Linux based systems increase GPL violations
- Today, many more people and companies unfamiliar with the history and values of Free Software start using and (re)distributing FOSS

Business Risk of GPL Violations

Or: How to convince your managers

If you ship a product that is incompliant to the GNU GPL,

- you are committing a copyright infringement not different from shipping a product with unlicensed copies of MS Windows
- you can face civil and criminal charges in court
- civil charges include (German jurisdiction)
  - immediate cease + desist (halt of product sales)
  - information of which quantity of the product has been sold to whom
  - damages for lost revenue (see dual licensing)
- civil charges can also be filed against every distributor/store/importer

Outlook

Outlook

- Blatant GPL violations in embedded devices are declining, but are likely to continue due to lack of skill or negligence.
- We'll see more *derivative works* types of GPL violations, and we'll see actual legal enforcement and precedent in this area over the next years.
- Stronger copyright protection demanded by content industry will also mean stronger protection for FOSS licenses. Imagine GPL enforcement with *three strikes* law in France ?!?