GPL Workshop
How to (not?) use Free Software

by

Harald Welte <hwelte@hmw-consulting.de>

Contents

Introduction

Who is speaking to you?

- an independent Free Software developer
- who earns his living off Free Software since 1997
- who is one of the authors of the Linux kernel firewall system called netfilter/iptables
- who has started gpl-violations.org to enforce license compliance
- who IS NOT A LAWYER

Disclaimer

Legal Disclaimer

- All information presented here is provided on an as-is basis
- There is no warranty for correctness of legal information
- The author is not a lawyer
- This does not comprise legal advise
- The authors' experience is limited to German copyright law

Ideas and Goals of the GNU GPL

- Free Software
  - Software that has fundamental freedoms:
    - to use it for any purpose
    - to "help your neighbour" (i.e. make copies)
    - to study it's functionality (reading source code)
    - to fix it myself (make modifications and run them)
- Copyleft
  - Is the legal idea to
    - exercising copyright to grant the above freedoms
    - assure that nobody can take away the freedom
- The GNU General Public License
  - Is a legal instrument to apply they copyleft idea on software

The GNU GPL Revisited

Revisiting the GNU General Public License

- Regulates distribution of copyrighted code, not usage
- Allows distribution of source code and modified source code
  - The license itself is mentioned
  - A copy of the license accompanies every copy
- Allows distribution of binaries or modified binaries, if
  - The license itself is mentioned
  - A copy of the license accompanies every copy
  - The complete source code is either included with the copy (alternatively a written offer to send the source code on request to any 3rd party)

Complete Source Code

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

- For standard C-language programs, this means:
  - Source Code
  - Makefiles
  - compile-time Configuration (such as kernel .config)
- General Rule:
  - Intent of License is to enable user to run modified versions of the program. They need to be enabled to do so.

Derivative Works

- What is a derivative work?
  - Not dependent on any particular kind of technology (static/dynamic linking, dlopen, whatever)
  - Even while the modification can itself be a copyrightable work, the combination with GPL-licensed code is subject to GPL.
  - As soon as code is written for a specific non-standard API (such as the iptables plugin API), there is significant indication for a derivative work
  - This position has been successfully enforced out-of-court with two Vendors so far (iptables modules/plugins).

Derivative Works

- Binary-only kernel modules
  - In-kernel proprietary code (binary kernel modules) are hard to claim GPL compliant
  - Case-by-case analysis required, as the level of integration into the GPL licensed kernel code depends on particular case
  - IBM is in the process of getting rid of all binary-only kernel modules. There are exceptions, but they are very clear ones (such as a filesystem port to linux, where the filesystem code already existed under another OS)
  - There is no general acceptance or tolerance to binary-only kernel modules in the Linux (development) community. Not even Linus himself has ever granted an exception for such modules!

Derivative Works

- Glue Code
  - Acts as glue layer between GPL licensed code and proprietary code
  - Some Vendors think they can avoid the GPL by doing so
  - Is definitely not a bullet-proof legal solution, especially when it is clearly visible that the only purpose of this glue code is to "get rid" of the GPL.

Derivative Works

- Moral Issues
  - Apart from what is legally possible, there are moral issues
  - Even if in a particular case there is no legal way to claim a binary-only kernel module is a derivative work, you might still be acting against the authors' wishes
  - By shipping binary-only kernel modules, you violate the "moral code of conduct" of the Free Software community
  - But it is the work of this very community that enables you to build your product based on Free Software
  - Such action might have long-term detrimental effects on the motivation of FOSS developers (dissatisfaction, demotivation, ...)

Collective Works

"... it is not the intent .. to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works ..."

- GPL controls "collective works"

"... mere aggregation of another work ... with the program on a volume of a storage or distribution medium does not bring the other work und the scope of this license"

- GPL allows "mere aggregation"
  - like a general-porpose GNU/Linux distribution (SuSE, Red Hat, ...)
  -

GPL And Embedded Systems

- Historical background:
  - The GPL was written for userspace programs running on existing operating systems
  - Covering a whole OS (and even userspace programs) is not an ideal match, but if you read it carefully it still makes sense

- Toolchain:

"... the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable."

  - Practical case:
    - You've modified gcc for a specific embedded platform
    - Therefore, this gcc is not "normally distributed with the operating system" and you have to distribute it together with the source code
    - gcc itself is covered under GPL, so you need to provide binaries and source code(!)

GPL And Embedded Systems

- The "Scripts"
  - (scripts to control compilation and installation, see earlier slide)
  - In case of embedded hardware, the "scripts" include:
    - Tools for generating the firmware binary from the source (even if they are technically no 'scripts')

- Embedded DRM
  - Intent of License is to enable user to run modified versions of the program. They need to be enabled to do so.
  - Result: Signing binaries and only accepting signed versions from the bootloader (without providing the signature key or a possibility to set a new key in the bootloader) is not acceptable!

Practical Source Code Offer

- Some Rules
  - The "complete corresponding source code" has to be made available
  - It has to be made available for each and every object-code version that was distributed
  - If you strip down the source code offer (e.g. remove proprietary source code), try to see whether the result actually compiles
  - If the product is mixed free / proprietary software, consider including the proprietary parts (as object code) in the "source code package", so the full firmware image can be rebuilt without having to tear apart an existing image and ripping out those proprietary programs from there.

The biggest myths about the GPL

The biggest myths about the GPL

- The GPL is not enforcible
- Software licensed under GPL has no copyright
- Unmodified distribution does not require source code availability
- The vendor can wait for a source code request (without offering it)

The most common mistakes

The most common mistakes

- not even once reading the GPL text and/or the FAQ from the FSF
- not including the GPL license text with the product
- not including a written offer with the product
- not considering that the GPL also applies to software updates
- only providing original source code (e.g. vanilla kernel.org kernel)
- not including the "scripts to control installation"
- only providing off-site hyperlinks to license and/ore source code
- not responding to support requests for source code
- charging rediculously high fees for physical shipping of source code

License Compatibility

- There's lots of Free Software available
  - Different Software uses different Licenses:
    - Linux: GPL
    - glibc: LGPL
    - apache: Apache Software License
    - Perl: Artistic
    - ucd-snmp: BSD
  - If you combine (i.e. link) differently-licensed software,
    - check license compatibility
    - in case of doubt, ask legal person and/or contact software authors
    - authors might give you an exception or consider making licenses compatible

Dual Licensing

- The copyright holder (often the original author) can provide alternative licensing
- Some projects do this as a business model (reiserfs, MySQL)
- In some projects it's impossible due to the extremely distributed copyright (e.g. Linux kernel)
- However, in smaller projects it never hurts to ask whether there would be interest in providing an alternative (non-copyleft) licensing

The End

- Further reading:
- The http://gpl-violations.org/ project
- The Free Software foundation http://www.fsf.org/, http://www.fsf-europe.org/
- The GNU Project http://www.gnu.org/
- The netfilter homepage http://www.netfilter.org/