gnu.org

Why Upgrade to GPLv3 - GNU Project

by **Richard Stallman**

Version 3 of the GNU General Public License (GNU GPL) has been released, enabling free software packages to upgrade from GPL version 2. This article explains why upgrading the license is important.

First of all, it is important to note that upgrading is a choice. GPL version 2 will remain a valid license, and no disaster will happen if some programs remain under GPLv2 while others advance to GPLv3. These two licenses are incompatible, but that isn't a fundamental problem.

When we say that GPLv2 and GPLv3 are incompatible, it means there is no legal way to combine code under GPLv2 with code under GPLv3 in a single program. This is because both GPLv2 and GPLv3 are copyleft licenses: each of them says, “If you include code under this license in a larger program, the larger program must be under this license too.” There is no way to make them compatible. We could add a GPLv2-compatibility clause to GPLv3, but it wouldn't do the job, because GPLv2 would need a similar clause.

Fortunately, license incompatibility matters only when you want to link, merge or combine code from two different programs into a single program. There is no problem in having GPLv3-covered and GPLv2-covered programs side by side in an operating system. For instance, the TeX license and the Apache license are incompatible with GPLv2, but that doesn't stop us from running TeX and Apache in the same system with Linux, Bash and GCC. This is because they are all separate programs. Likewise, if Bash and GCC move to GPLv3, while Linux remains under GPLv2, there is no conflict.

Keeping a program under GPLv2 won't create problems. The reason to migrate is because of the existing problems that GPLv3 will address.

One major danger that GPLv3 will block is tivoization. Tivoization means certain “appliances” (which have computers inside) contain GPL-covered software that you can't effectively change, because the appliance shuts down if it detects modified software. The usual motive for tivoization is that the software has features the manufacturer knows people will want to change, and aims to stop people from changing them. The manufacturers of these computers take advantage of the freedom that free software provides, but they don't let you do likewise.

Some argue that competition between appliances in a free market should suffice to keep nasty features to a low level. Perhaps competition alone would avoid arbitrary, pointless misfeatures like “Must shut down between 1pm and 5pm every Tuesday”, but even so, a choice of masters isn't freedom. Freedom means *you* control what your software does, not merely that you can beg or threaten someone else who decides for you.

In the crucial area of Digital Restrictions Management (DRM)—nasty features designed to restrict your use of the data in your computer—competition is no help, because relevant competition is forbidden. Under the Digital Millennium Copyright Act and similar laws, it is illegal, in the US and many other countries, to distribute DVD players unless they restrict the user according to the official rules of the DVD conspiracy (its web site is http://www.dvdcca.org/, but the rules do not seem to be published there). The public can't reject DRM by buying non-DRM players because none are available. No matter how many products you can choose from, they all have equivalent digital handcuffs.

GPLv3 ensures you are free to remove the handcuffs. It doesn't forbid DRM, or any kind of feature. It places no limits on the substantive functionality you can add to a program, or remove from it. Rather, it makes sure that you are just as free to remove nasty features as the distributor of your copy was to add them. Tivoization is the way they deny you that freedom; to protect your freedom,

GPLv3 forbids tivoization.

The ban on tivoization applies to any product whose use by consumers is to be expected, even occasionally. GPLv3 tolerates tivoization only for products that are almost exclusively meant for businesses and organizations.

Another threat that GPLv3 resists is that of patent deals like the Novell-Microsoft pact. Microsoft wants to use its thousands of patents to make users pay Microsoft for the privilege of running GNU/Linux, and made this pact to try to achieve that. The deal offers rather limited protection from Microsoft patents to Novell's customers.

Microsoft made a few mistakes in the Novell-Microsoft deal, and GPLv3 is designed to turn them against Microsoft, extending that limited patent protection to the whole community. In order to take advantage of this protection, programs need to use GPLv3.

Microsoft's lawyers are not stupid, and next time they may manage to avoid those mistakes. GPLv3 therefore says they don't get a “next time”. Releasing a program under GPL version 3 protects it from Microsoft's future attempts to make redistributors collect Microsoft royalties from the program's users.

GPLv3 also provides users with explicit patent protection from the program's contributors and redistributors. With GPLv2, users rely on an implicit patent license to make sure that the company which provided them a copy won't sue them, or the people they redistribute copies to, for patent infringement.

The explicit patent license in GPLv3 does not go as far as we might have liked. Ideally, we would make everyone who redistributes GPL-covered code give up all software patents, along with everyone who does not redistribute GPL-covered code, because there should be no software patents. Software patents are a vicious and absurd system that puts all software developers in danger of being sued by companies they have never heard of, as well as by all the megacorporations in the field. Large programs typically

combine thousands of ideas, so it is no surprise if they implement ideas covered by hundreds of patents. Megacorporations collect thousands of patents, and use those patents to bully smaller developers. Patents already obstruct free software development.

The only way to make software development safe is to abolish software patents, and we aim to achieve this some day. But we cannot do this through a software license. Any program, free or not, can be killed by a software patent in the hands of an unrelated party, and the program's license cannot prevent that. Only court decisions or changes in patent law can make software development safe from patents. If we tried to do this with GPLv3, it would fail.

Therefore, GPLv3 seeks to limit and channel the danger. In particular, we have tried to save free software from a fate worse than death: to be made effectively proprietary, through patents. The explicit patent license of GPLv3 makes sure companies that use the GPL to give users the four freedoms cannot turn around and use their patents to tell some users, “That doesn't include you.” It also stops them from colluding with other patent holders to do this.

Further advantages of GPLv3 include better internationalization, gentler termination, support for BitTorrent, and compatibility with the Apache license. All in all, plenty of reason to upgrade.

Change is unlikely to cease once GPLv3 is released. If new threats to users' freedom develop, we will have to develop GPL version 4. It is important to make sure that programs will have no trouble upgrading to GPLv4 if and when we write one.

One way to do this is to release a program under “GPL version 3 or any later version”. Another way is for all the contributors to a program to state a proxy who can decide on upgrading to future GPL versions. The third way is for all the contributors to assign copyright to one designated copyright holder, who will be in a position to upgrade the license version. One way or another, programs should provide this flexibility for future GPL versions.