Free Software Distributions and Ancillary Rights

Eben Moglen & Mishi Choudhary

March 27, 2017

Distributions and Their Legal Structures

Distributions of free software involve sharing of computer program, which is mostly governed by copyright law.[1] But other legal rights, involving trademark, patent, trade dress protection, protection against unfair competition, and other legal doctrines are potentially involved as well. At the level of a single work, such as the Apache web server or the GNU Compiler Collection, these ancillary legal rights rarely raise complex issues of nesting or interaction for analysis.

But when hundreds or thousands of programs and associated files containing documentation or configuration data combined into "packages" are then aggregated into "distributions" such as Debian, Fedora, RHEL or Ubuntu, the significance of these related rights increases, and the complexity of their interaction does as well. How the trademark and other rights associated with the component packages—those of the Apache Software Foundation on the packages it publishes, for example— interact with the trademarks, competition rights and other ancillary powers of the distribution manufacturers, such as Red Hat, Debian or Canonical, cannot be determined solely by analysis of the copyright licenses on the programs involved. Each party's trademark licensing policies, for example, will affect the real rights and relations of parties, beyond the content of the FOSS licenses used.

This document takes these "peripheral" rights as its central subject.

Copyrights

Anthologies

Copyright licenses are both the primary legal regulation of users' right in distribution components, and one of the ancillary rights bundle for distribution-makers.

[1] *See* Chapter 2 of A Legal Issues Primer for Open Source and Free Software Projects

The unit of copyright law is the "work," which usually signifies one computer program within a package, or one manual presented as a text file or series of text files. A package also usually contains some uncopyrightable works, configuration data or other material which lacks the required "originality" for copyrightability. But all of the files in a package selected and arranged by the distribution-maker, also form part of the complex combination of software that *is* the "distro."

Understanding the distribution, as a copyright matter, therefore requires discussion of compilation copyright. "Compilation" as a term of art in US copyright law means "a work formed by the collection and assembling of preexisting materials or of data that are selected, coordinated, or arranged in such a way that the resulting work as a whole constitutes an original work of authorship."[2] Compilations, under US law, includes "collective works," which are defined as "a work, such as a periodical issue, anthology, or encyclopedia, in which a number of contributions, constituting separate and independent works in themselves, are assembled into a collective whole."[3]

Under 17 U.S.C. §103(b),

> The copyright in a compilation or derivative work extends only to the material contributed by the author of such work, as distinguished from the preexisting material employed in the work, and does not imply any exclusive right in the preexisting material. The copyright in such work is independent of, and does not affect or enlarge the scope, duration, ownership, or subsistence of, any copyright protection in the preexisting material.

A free software distribution, like Debian, Fedora, RHEL or Ubuntu is a compilation, not only a collective work, because the packages within the distribution will themselves contain—along with various separate and independent works, consisting of computer programs and documentation—other, modified, copyrightable or uncopyrightable files that "coordinate" and "arrange" the software, such as configuration data and package management information. These additions, necessary to the integration of the component works, create an original work of authorship.

Copyright in a compilation gives the compiler exclusive rights to the selection and arrangement of the individual components, regardless of the copyright status of those components, and who holds copyright on them.[4] The compilation copyright neither ousts nor modifies the respective authors' exclusive rights, which must be licensed to the compilation-maker in order to create the compilation.

[2] 17 U.S.C. §101
[3] *Id.*
[4] Feist Publ'ns, Inc. v. Rural Tel. Serv. Co., 499 U.S. 340, 348-350 (U.S. 1991)

Selection and Arrangement

The activity of putting together a distribution of free software packages involves selecting and arranging packages; creating necessary intra-package and distribution-wide management and configuration information; compiling, processing or otherwise creating executable files from source code trees; and arranging the resulting distribution files for convenient acquisition, installation and updating by users. Some of these actions give rise to copyrightability in the resulting aggregation of software, in the same way that the choice and sequencing of literary works in an anthology gives rise to copyright.[5] The decision to include the Apache web server, or the lighttpd web server, or both, in a distribution is part of the selection and arrangement that gives rise to the "anthology copyright" on the distro as a whole. Other portions of the distribution-maker's activity result in purely functional files or mechanical transformations of others' copyrighted works, valuable to the user but not subject to copyright because not an "original work of authorship," such as the binaries compiled from others' source code. Some results from these latter activities may give rise to other ancillary rights, under, for example, domestic law of legal protection for databases created and maintained as part of the distribution.[6]

In-House Code

The process of making distributions involves no small amount of additional programming. Maintenance to package code, works specific to the distribution, activities and projects specifically produced or sponsored by the distribution-maker—all comprise a significant fraction of the software in major distributions. These activities produce their own independent copyrights in many instances. These work- or package-level copyrights form another part of the web of ancillary rights held by distribution makers.

Configuration copyright

Most packages in a distribution contain configuration data, specifying details of the package's operation when installed. Default options, file pathnames, device- or system-specific information and other functionally useful details are contained in these configuration files, which are often annotated by text explaining for human readers what the configuration values mean and how to modify them. These configuration files are thinly copyrightable, if they can be said to fall within copyright scope at all. The sum of these configuration files—which allow all the packages in a distribution to work together smoothly because they are compatibly configured—expresses precisely the creative selection and arrangement of the

[5] *See generally* eScholar, LLC v. Otis Educ. Sys., Inc., 387 F. Supp. 2d 329 (S.D.N.Y. 2005)

[6] Choice of law issues are not within the scope of this discussion. For present purposes, the analysis is conducted solely under US copyright and trademark laws.

distro. So while the copyrightability of the individual files may be doubtful or scant, the overall effect of these fragmented copyrights is substantial.

Nested Copyright

Making distributions out of others' free software packages therefore results in "nested" or concentric copyright. The copyrights on files and works within packages are licensed individually, according to the FOSS or free culture licenses on the programs and documentation contained. Where these license terms offered are copyleft terms, they are essentially the only terms that apply to the licensed works no matter what happens at outer concentric layers: no enclosing copyright may be used to place additional restrictions on those works.[7] Where the terms offered on works are "permissive" in nature, license terms at outer concentric layers *may* change the terms applicable to those component works. For example, a distribution containing thousands of packages may be licensed overall by a EULA that conditions use on acceptance of a warranty disclaimer. If the constituent copyright licenses on individual works within the distribution are permissive, that disclaimer may affect rights in the constituent works. When the disclaimer in the EULA wraps a copyleft license on a constituent work, however, the component license may either abrogate the EULA's disclaimer, as do the terms of GPLv2, or permit such a disclaimer for purposes of compliance with local domestic law, as do the terms of GPLv3. In this situation of "nested terms," permissive licensing propagates copyright permissions "from outside inwards," while copyleft licensing propagates the permissions "from inside outwards." A distribution maker's power to enforce its exclusive rights over the compilation is therefore limited in the case of free software distributions, in two directions. First, the compilation copyright extends only to the unique elements of selection and arrangement. It is not possible, for example, to say that any distribution with kernel packages, a C library package and userland applications packages infringes copyright on the selection and arrangement of one particular distribution. Second, the presence of copyleft-licensed components within a distribution limits the restrictions that can be placed by distribution-wide licensing on at least some of the component works within the distribution.

For the distribution-maker, the copyrighted compilation includes both source code files and executables. The binary files within the distribution will be copylefted if they are made from GPL'd sources in whole or part, but if the binaries were produced from the compilation of permissively-licensed source code—or source code licensed under "weak copyleft" terms like those in CDDL, Eclipse and similar licenses—those binaries may be subject to the copyright terms of the distribution-wide EULA or other outer-layer licenses, thus forming an additional ancillary right for the distribution-maker.

[7] *See* Concepts and License Mechanics of Copyleft in Software Freedom Law Center Guide to GPL Compliance 2nd Edition

Another form of distribution or aggregation of free software has become important of late. A "container," of whatever particular format now fashionable it may be, is an aggregation of free software comprising a "ready to run" filesystem containing OS components, other relevant dependencies, and the container user's own application workload. Within the container, these various software aggregations may be represented as a collection of compressed archive files, for example, all of which, when expanded, present the full filesystem intended for execution. The container can be compared roughly to a "live CD" filesystem, or a bootable USB filesystem, which have been and remain popular modes of aggregate free software distribution.

In a container, as in one of these conceptual predecessors, the copyleft binary components are—from the point of view of GPLv2, for example—placed in "mere aggregation" with other non-GPL binaries, not in a combination triggering copyleft. So being in the same container with a GPL'd binary doesn't propagate the copyleft "outward." (The same requirements to provide or offer source code for the GPL'd executables apply, however, if A provides a "container" to B as would be imposed if A provided a CD or USB key to B with the same GPL'd executables on it.)

But the selection and arrangement of works aggregated in a container could, even if the container was made under program control, give rise to "anthology" copyright over the aggregate. The terms on which this anthology is licensed could theoretically modify the terms on permissively-licensed works within the container, as we have seen in other contexts.

So far as we are aware, no container-related anthology-level copyright issues have so far occurred in practice. Parties have been more concerned with a non-existent copyleft inheritance issue, where GPL'd binaries are included in containers also containing proprietary or other non-copyleft executables. Once this uncertainty has subsided, however, the issue of EULA-like enclosing licenses on containers is likely to arise in one or more commercial contexts.

Trademarks

The essence of trademark is the association of a name with the source or origin of goods in the marketplace.[8] The property right in the name given to the trademark holder is balanced by the social advantage consumers derive from the assurance of quality names convey. Accordingly, in the United States, the essence of trademark liability is the creation of "confusion" by disturbance of the relationship between the known mark and the inferred source, origin, and quality of goods.[9]

[8] 15 U.S.C. §1127
[9] 15 U.S.C. §1114

Descriptive names can be trademarks, in the US, only to the extent that the descriptive name has acquired "secondary meaning," that is, has come to be associated with the particular source or origin of goods through use and acceptance in the marketplace. Even an arbitrary name, such as Kleenex, can become "genericized" through the process of broad social use, such that the secondary meaning associated with a particular producer of paper in which to blow your nose has evaporated over time. Unlike copyrights, trademark rights are subject to defeasement through failure to protect the mark. The trademark owner's failure to prevent confusion can lead to loss of rights.[10] Trademark owners are therefore required to take measures to "police" their marks, which is why trademark enforcement often occurs in situations that seem in isolation oppressive or trivial.

Free software programs and packages are named; the names chosen are often registered as trademarks.[11] In order to avoid possible loss of the mark through a finding of "naked licensing," the projects using trademarked names need to make licensing policies, stating clearly the terms on which the marks, whether textual or graphical, may be applied to modified or redistributed versions. They need to be prepared to show that their policies are enforced.[12] Unlike commercial licensors, who belligerantly enforce their marks, it is often in the interest of free software projects to make liberal trademark policies, allowing those downstream who modify or repackage their code to make use of their marks. The origin of the free software is explicitly designated, by the use of the package's trademarked name, while the fact that the distribution and some or all component packages have been modified is also indicated in the modified version's documentation and source code. Clearly indicating the origin of modifications while retaining the name of the package obviates risk of confusion.

Distributions, in turn, also have names, which are very likely to be registered as trademarks. They too have licensing policies, which govern the conditions under which the names can be used to identify, among other things, compilations of software that originate from the trademarked distribution, but which may have been modified downstream. Community distributions, such as Debian, are likely to have different trademark licensing policies than commercial distributions. The interests of the two kinds of parties dictate different balances between encouraging modification and redistribution, on the one hand, and protecting asset values, including of intangible property rights in names, on the other.

Our practice at SFLC involves writing and enforcing licensing policies for distributions like Debian, as well as package trademarks like those of Kodi (formerly XBMC). We therefore deal with the consequences of "nested trademarks" as we deal with the consequences of nested copyright. Here, the effect of nesting is different than it is in copyright. In the nesting of copyrights, outer-level licenses, like EULAs on an entire distribution, can displace terms at the inner

[10] *See, e.g.,* Bayer Co. v. United Drug Co., 272 F.505 (S.D.N.Y. 1921)
[11] *See* Chapter 5 in A Legal Issues Primer for Open Source and Free Software Projects
[12] FreecycleSunnyvale v. The Freecycle Network, 626 F.3d 509 (9th Cir. 2010)

level, unless those work-level or package-level terms are protected by copyleft. In trademark, however, the terms for use of a distribution name have no effect on the terms governing the use of package or program trademarked names. The policy regarding the use of the mark "Debian," or the mark "Ubuntu" cannot displace or modify the policies regarding use of the words "Apache," or "Kodi," put in place by the owners of those package-level marks.

The Problem of Origin

In free software distributions, where modifications occur at many levels as code moves from upstream to multiple downstreams, the origin of goods cannot necessarily be referred to a single point. The packages contained within the distribution have an origin indicated by the project name associated with the package: the Apache webserver, GNU tar, etc. The distribution name indicates another point of origin: the Debian distribution's sid version's modified copy of the Apache webserver, for example. In some cases, the trademark licensing policy of the component will dictate that the name be changed in the distribution, as with the historical renaming of Firefox to Iceweasel in Debian.[13]

At each level of enclosure, from the single program through the package to the distribution, the interest of the trademark holder is in preventing uses of the name which may confuse the ultimate user about the origin of the software, both in order to protect the user's quality perception of the "brand" established by the name, and to prevent harmful associations. The latter motive is important, for example, when a media player like Kodi is distributed by makers of "distributions" who add plug-ins to the player that facilitate the acquisition of copyright-infringing media files.

For this reason, the trademark licensing policies of both package and distribution markholders may require that materials bearing trademarks (graphical elements in UI design, documentation, etc., for example) must be removed from modified and redistributed versions, to prevent "confusion" as to origin. If such requirements are imposed by policy, but made difficult to implement, they could function as barriers to the redistribution of the free software itself, which—in the case of copylefted software—would imply a prohibited imposition of additional restrictions. In such cases, copyleft requires that the materials added midstream by the trademark owner to preexisting copylefted software be so segregated that they can with reasonable effort be removed by downstream modifiers. Nothing, on the other hand, ever *requires* a downstream party to retain a trademarked name it would rather remove.

[13] As of February 2016, issues between the Debian project and Mozilla Foundation about use of the Firefox trademark, pending since 2006, have been resolved.

Confusion Prevention—Package Quality Control

The rules that determine what names can be used for modified versions of FOSS programs are found not in the copyright licenses, but in the trademark licensing policy of the entity that controls each mark. The licensor's primary incentive is to prevent the quality assumptions that have been built into the mark from being deprecated downstream. This incentive prevents projects and packagers from being as permissive in trademark licensing as they are in their copyright licenses. It is sufficient, in most free software copyright licenses, that modified versions be so designated [14] But if program FOO, whose name is trademarked by its authors, is modified downstream and the modifications are properly designated in the source code as required by FOO's license, that does not put at ease the minds of FOO's developers if the modifications contain serious bugs that will cause users of modified FOO to experience harm, thus deprecating FOO's reputation for quality. For this reason, the FOO project's trademark policy could require that modification patchsets be reviewed by FOO's developers before they will license use of the trademark FOO as the name of the modified version.

Confusion Prevention—Distribution Conflict

Trademark licensing policies can also play a role in avoiding confusion not only in end-user perception but also in technical coordination at the distribution level. Let's take an imaginary trademarked free software distribution called Blue Sock, containing the Linux kernel, a C library, and a collection of userland applications, some produced by third-party free software projects, but with a large component of in-house code. BS is arranged and configured in a distinctive fashion, optimized for use in retailing and agriculture. Downstream, Blue Sock is modified by Cloudiness Worldwide, a "virtual personal server" vendor to aquaculturists everywhere. CW may apply patches to packages within the Blue Sock distribution it offers to CW customers that conflict with maintenance directly applied by BS upstream, or which may interact with BS's in-house code in unexpected ways. Resulting breakage in Cloudiness VPSs may be wrongly attributed by end users to quality failures at Blue Sock. If Blue Sock is a commercial distribution, this risk may well be sufficiently grave to justify trademark policy restrictions that would prevent Cloudiness Worldwide from calling its VPS operating system "Blue Sock" unless, for example, BS is accorded an opportunity to review and coordinate its technical maintenance activities with CW's.

[14] *See* Managing copyright information within a free software project

Communities and Companies

The difficulties that arise from overlapping rights occur most sharply at the transitions between non-commercial and commercial parties. For non-commercial projects, packagers, or distribution-makers—who as a general rule do not license rights for revenue—license terms on trademarks are set by the need to affirm quality and provenance. Commercial parties, both downstream and upstream from non-commercial producers and packagers, on the other hand, not only may wish to monetize their ancillary rights, but also to use those rights to compete more effectively against other parties in the marketplace. As a result, the interleaved structures of nested rights created by aggregating free software programs into distributions may behave counter-intuitively when parties within the structure aggressively assert their ancillary rights.

Although the same issues present at commercial/non-commercial boundaries are present everywhere, the variance in motives leads to greater chance of mutual misunderstanding and resulting social conflict.

As we have observed, copyright and trademark rules "nest" differently, in the concentric structures of free software distribution. On the trademark side, enforcement of trademark licensing policies is legally required. But non-commercial upstreams not engaged in monetizing their rights tend to have more liberal trademark licensing policies than their commercial downstreams. Thus, for example, Debian allows far more use of the Debian mark by downstream distributions than Red Hat allows. The Red Hat trademark policy does not allow even unmodified copies of RHEL to be called "Red Hat" by the distributor, without specific written permission. Debian's policy, on the other hand, allows any factually truthful assertion of Debian's relation to other parties' services and software, so long as no misleading or confusing use of the Debian marks is involved. So a modified distribution of software made downstream from Canonical using modified Ubuntu packages could conceivably be called "Debian-based," in compliance with Debian's trademark policy, even though it could not be called "Ubuntu" under Canonical's policy.

More generally, if non-commercial distribution A, with a liberal trademark licensing policy, is upstream from commercial distribution B, which chooses to impose more restrictions on the use of its own name in support of its business interests, the makers of downstream distributions or providers of PaaS services C, D, and E may choose to trumpet their descent from A, dropping B's name from sight altogether, to B's ultimate loss. The resulting complexities in both policy drafting and enforcement strategy have no analogues in conventional trademark practice.

The copyright rules applicable to the same situation, on the other hand, are determined by the right/left valance of the licensing of each component work. If, for example, Canonical has applied a EULA that imposes requirements on binaries made from permissively-licensed free software programs distributed

as part of Ubuntu and also in Debian, those binaries cannot be copied into another distribution under non-Canonical terms. Component packagers who use permissive licenses may be surprised to discover that the terms on which downstream users receive their unmodified packages may not be the ones they themselves applied. Executables of programs subject to strong copyleft cannot be similarly restricted, because the copyleft on the binaries prohibits the imposition of any additional restrictions.

Further problems can arise from the enclosing EULAs, or services agreements, covering PaaS or SaaS distributions of FOSS stacks. The EULA or terms of service agreement may replace or supplement the terms of every permissive license used at the package or program level within the distribution, and the terms on every binary produced from source code under semi-copyleft licenses like Eclipse or CDDL. If they do not explicitly exempt copyleft-licensed components from their terms, additional restrictions are imposed on the copylefted components, and infringement results. The most subtle form of such infringement may be a term in the services agreement that "licenses" the software in use to the user. Such a clause would be predictably present if the services agreement involved the use of proprietary software, but it falls foul of the prohibition of sublicensing in GPLv3 §2 and the prohibition of additional restrictions in both GPLv2 §4 and GPLv3 §10. Any award of preliminary injunctive remedies during litigation of resulting claims by even one component's copyright-holder may effectively halt the entire distribution. GPL, in both version 2 and version 3, attempts to save downstream users who are themselves complying with copyleft from being adversely affected by such litigation. But the licenses can only have their desired effect on behalf of those users who have received a distribution or to whom software has been conveyed. Where no distribution or conveyance has happened, as in the PaaS and SaaS contexts, end users can have no protection against the harm accruing from their upstream provider's mistake in sublicensing them.

Conclusion

As this analysis shows, the rights acquired by distribution-makers outside the copyright licensing context of individual programs and packages are significant, both in the sense that they provide ample opportunities for distributions' terms to affect downstream businesses, and in causing potentially unexpected legal issues for both upstream and downstream parties.

The invocation of compilation copyright, and its licensing through EULAs and terms of service agreements by PaaS and SaaS vendors, is an outcome consistent with application of standard copyright doctrine in the FOSS context. Such compilation licenses may alter the terms of permissively-licensed packages in ways that can render executables made from permissively-licensed source code no longer freely redistributable, even without modification. Trademark licensing policies applied at the distribution level cannot similarly modify terms on names

associated with upstream parties, but they can effectively prevent reuse under commercially-necessary terms by cloud services providers. Terms of service that purport to license PaaS and SaaS offerings may interact with copyleft licenses in ways that harm downstream users despite terms in those copyleft licenses designed to hold downstreams harmless in the event of license violation by a mesne distributor. For legal practitioners working on FOSS, although the program- or package-level licensing details will always be foregrounded, careful attention to the employment and enforcement of ancillary rights is unavoidably necessary.