Podcast Transcript

Open Source is Everywhere: A Primer for Compliance

Date: February 17, 2021

Guest: Timothy D. Casey **Host:** Amy Kattman

Run Time: 15:22

For questions and comments contact:

Timothy D. Casey

Partner
Seattle
T: 1.206.332.1107 | tcasey@bakerlaw.com

Kattman: Most software being developed today includes open-sourced license components, but many developers and their lawyers are unaware of the implications and requirements of utilizing such components. I'm Amy Kattman and you're listening to BakerHosts.

On today's episode we will discuss a range of open-source license types including understanding the pluses and minuses of each and how to comply with their notice and attribution requirements. A special tip regarding the copyright registration of one's own code that includes open source will also be provided. To provide us with insight into this topic we have Tim Casey, a partner in the Intellectual Property Group at BakerHostetler. Tim has a background in electrical engineering and has written and prosecuted thousands of patents worldwide. Welcome to the show Tim.

Casey: Happy to be here. Looking forward to an enthralling podcast.

Kattman: Well, Tim, to begin can you explain what is open-source, and how it's different from public domain?

Casey: Sure. So, if software is truly in the public domain that means that nobody has any ownership rights to that cut or if they did some point at time, because they must have if they'd written it and had a copyright, they've expressly dedicated those rights to the public domain. In other words that means that that code is free to be used by anybody, copied, revised, sold, incorporated into other software, you can

do anything you want to with it without any restriction or without having to pay a renumeration to anybody for it. But open-source code on the other hand is not public domain, and that means someone owns rights in the code, and they're using those rights to limit what other people can do with that software. Even if it's freely available or open to the public to use, there are still restrictions associated with it. And the vehicle that you use to exercise those rights in their open-source license has to be agreed to as part of the right to use that software, and we call those open-source licenses.

Kattman: So, Tim, what are the different types of open-source license?

Casey: So, there's three primary types of open source. There's copyleft, weak copyleft, and permissive. And there's a bunch of different variations or permutations of each of those, but those are the three main areas. And copyleft is probably the most well-known, or at least many people have heard of it at some point in time. It was a concept largely popularized by an American software programmer named Richard Stallman, who I once met, interesting guy. And he initially promoted what's called free software, which was free, meaning that you didn't have to pay for it, but which was subject to a number of restrictions, and copyleft was a version of that. Stallman later went on to establish something called the Gnu Project. And that's new is spelled G-N-U like the animal, which is a recursive acronym for GNU's Not Unix, because the whole point behind that project was to develop an operating system that had all the useful aspects of Unix in it but wasn't Unix.

The basic premise of copyleft is that you're using a copyright as a legal mechanism to protect the modification or redistribution of what would otherwise be free software. So, when someone incorporates a component that's licensed under a copyleft license such as GPL version 2 or version 3, they must release its source code as well as the rights to modify and distribute the entire code that they're making available to license that source code under the same GPL license that they received the component in. And that's true even if the licensed component, which is the minor portion of the entire code, I mean it could be less than 1%. As long as you've got GPL license code in there then all of the software is subject to GPL.

And there's lots of other terms and conditions that apply as well in the GPL license. For example, the resulting code is considered a work based on GPL and therefore you can't claim patents in it, you can't copyright the software, and you can't do anything to restrict anyone else rights to use that software. So, you have to display a copyright notice, you have to provide a disclaimer of warranties, you have to include the intact GPL notices and a copy of GPL, and you can't change anything or add any additional terms to the license that you provide for your own code. And these basic terms are consistent across all the different versions of GPL that have been published over the years.

There has been some effort in more recent years, like in version 3, to make it more compatible with other, so, they could be combined with other licenses like more permissive licenses, and I'll talk about that in a minute. But, while copyleft

license remains popular in some domains, the restrictions that are associated with them have really caused their adoption to wane over time.

Kattman: How about weak copyleft?

Casey: So, weak copyleft is, it's just another version of copyleft. But it typically allows distributors to link licensed software with non-free programs, which thereby enables the entire code to be sold instead of having to be given away for free as long as the copyleft portion of the code, in its original or modified form, is still made public under the same terms as the original. So, if you were going to use a weak copyleft license, you'd want to make sure that the portion that is subject to that license is a separate component that could be broken out from the rest of the code. And that way you can make that available without having to make all the rest of it available.

Kattman: Okay and finally, could you talk about permissive?

Casey: Sure. Permissive open-source licenses place minimal restrictions on how others can use the open-source components. The most common permissive open-source licenses are the MIT and Apache 2.0 licenses, and I'll talk more about those in a minute. With these types of licenses allow varying degrees of freedom of use to modify and to redistribute the open-source code, including permitting its use in derivative works, which can be proprietary, meaning that you can sell them, etc., and restrict someone else's use of them. And basically, they require nearly nothing in return, which again, I'll talk about.

Kattman: Can you tell us what direction open-source licensing is trending?

Casey: Well, definitely away from GPL and copyleft and towards permissive licenses. So, as of last year, about 67% of all open-source components that were available in the market were permissive licenses, which is up about 3% over the prior year, and only about 33% are copyleft, either weak or strong of some type. But even as recently as 2012, copyleft licenses were a majority of the open-source licenses, representing about 59% overall. So, things have definitely been trending towards permissive licenses.

Open-source has become a mainstream primarily because it has so few strings attached to it because a number of big software companies, like Microsoft and Google, have supported a number of major open-source projects. The most popular permissive license is the MIT license, which represents about 27% of all open-source licenses. And that's partly because the license is short and it's relatively easy to understand, unlike the GNU Affero license or the GNU GPL with Classpath exception license. These are complicated to read. Even I as a lawyer, so a lot of experience in dealing with this for going on 25 years, find the language in the new licenses to be confusing. So, I can imagine how someone who hasn't spent a lot of time looking at these would just read it and not be able to figure it out.

The MIT license, it doesn't have that issue. It's very simple. On the other hand, and it simply tells downstream users what they can't do, requires them to publish a copyright notice, and disclaims implied warranties, and that's about it. The Apache 2.0 license is also popular, it's about 23 percent of the market. Its main conditions are that the copyright and license notices have to be preserved, and that an express grant to any patent rights must be provided. Other than that, though, the license allows licensed works, modifications, and larger works to be distributed under different terms, and without having to make any source code available. So, these are very popular for those reasons.

Kattman: Thank you. Let's talk about requirements. What are the most common requirements of open-source licenses?

Casey: Well, as I, the one common trend you saw in, or you heard, in all of the ones that I've discussed above is that you have to provide the copyright notice and some attribution, so where the software is coming from, and even though that seems like a minor requirement, that's the thing that I find many people fail to abide by. So, they incorporate the code and then they don't incorporate the copyright notice and attribution requirements. Which means, basically, that you're not in compliance with the licensing, you've done that. The other really common provision is buyer beware or as is for warranty provisions, and those are important because the person licensing the code to you to use, wants to make sure that if something goes wrong with your software and it causes problems for someone else, they don't have any liability for it. Especially since they haven't gotten paid, they want to make sure that they're not going to be on the hook for some of these.

Kattman: Right. Now, why is it a good idea to audit your own operations?

Casey: Well, issues associated with your use of open source can come up unexpectedly in many cases. The use and compliance with open-source licenses, for example, is a common due diligence issue that's raised in many mergers and acquisitions and financing. So, you may find yourself suddenly being asked, well what open-source licenses do you have in your code and are you in compliance with the terms and you may have no idea. And you're running around, you're trying to talk to the software programmers and engineers about what's there. They may not know, or it can be very confusing, and if you have a hard time answering that questions, that's going to be a very clear sign that you've got some issues going on with that and you want to make sure that that's not going to be a problem, so you may not be anticipating a mergers or an acquisition or a financing any time soon, but they can come up unexpectedly and then you need to be prepared to deal with it. So, getting a clear understanding of what the open-source software has been used, in terms of those licenses, as well as sort of managing that on a going forward basis. It's a great thing to do to ensure that you're prepared to answer any inquiries that might come up, instead of scrambling around to try and assess and fix the problems in a hurry. It can also be an issue when you're commercially licensed, you're a software to others, because they may want to, it may not just be an application that's running on the desktop, you may want to license somebody to some component that you've made available and that's

revenue stream for you. They're going to be worried about what issues they're creating for themselves that they take your software. So there's a number of ways in which you can perform an audit, there's a variety of different entities out there that provide auditing services, like WhiteSource and Black Duck and other entities of that type, and people like me often help clients try and figure out how to deal with that kind of aspect.

Kattman: Tim, as a final question, can you copyright-register your own software if it includes open-source?

Casey: Well if you were listening carefully, I said if you have a GPL license in your software, then you can't copyright it. So, in that case, no. At least, not without violating the terms of the GPL license, but outside of GPL, you can still register a copyright your own software. But the key point there is that you didn't write the open-source code that's being utilized in your software, somebody else did, so you can't claim to be the author to that, so one of the things you have to do in your copyright registration is identify all of the open-source code that's included in that and it expressly disclaim any rights to that code in your application. This is another good reason to understand what open source has been used in your code and under what license terms so that you're in a position to actually protect your income. And this can be complicated, I've had a client recently with as many as 468 open-source components and their own code that we had to go through and identify and then disclaim as part of the copyright process, and it was complicated and it took a long time, therefore it cost a lot, and so that was quite the mess, but boy, that was a fun project.

Kattman: Well thank you for joining us today, Tim.

Casey: It's my pleasure, thank you for having me.

Kattman: If you have any questions for Tim, his contact information is in the show notes. As always, thanks for listening to BakerHosts. Comments heard on BakerHosts are for informational purposes and should not be construed as legal advice regarding any specific facts or circumstances. Listeners should not act upon the information provided on BakerHosts without first consulting with a lawyer directly. The opinions expressed on BakerHosts are those of participants appearing on the program and do not necessarily reflect those of the firm. For more information about our practices and experience, please visit bakerlaw.com.