Zack's Kernel News

Chronicler Zack Brown reports on the latest news, views, dilemmas, and developments within the Linux kernel community.

By Zack Brown

Licensing Confusion

An interesting little licensing escapade occurred recently. Matthew Garrett noticed that CloudLinux corporation had released their lve kernel driver under a new, proprietary license, whereas before, it had been released under the GPL. Technically, nothing too fishy is going on here because the original author of GPL'd code is still free to release that code under other licenses. But Matthew noticed that, in this particular case, the driver still pretended to the kernel to be a GPL'd driver and still used kernel symbols that were restricted to GPL-only code.

That's a big no-no, and Matthew submitted a patch to make the kernel automatically treat the lve module as non-GPL, whether it said it was or not. But that wasn't the end of the story because licensing is a very touchy legal subject, and free software advocates want to make good and sure that the GPL does not become irrelevant. Greg Kroah-Hartman was particularly interested in following up with CloudLinux about this, partly because their module used GPL-only symbols that he himself had created. He was nonplussed.

The conversation continued, and Matthew's code to block the lve module was accepted into the kernel. But, at some point, the CEO of CloudLinux, Igor Seletskiy, wrote to the mailing list, saying:

I am very sorry about this situation. It was my oversight. We planned to close source the module, and we will do it later on. Yet, it looks like one of our developers missed the point – and did things incorrectly. Please, give us two-three weeks to straighten things out. By the end of three weeks I plan to have source RPMs with the GPLed version of the modules available in our source repositories. Later on we will have new module that is not GPL released. Once again – I am sorry about the incident. We haven't planned to deceive anyone. This was more of an internal miscommunication then anything else.

The discussion at this point petered out – the problem having been sufficiently explained. But Greg did argue that CloudLinux was not legally allowed to take two to three weeks to deliver the source code to a GPL'd binary. In fact though, the GPL version 2 says that one acceptable means of source distribution is to release the binary with "a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange." So apparently, a written offer would be good enough, and the three-week delay might be legally acceptable.

Kernel Development Overflow

Thomas Gleixner posed an interesting question – all the more interesting in that it was taken so seriously. He pointed out that interest in the kernel was continuing to grow and that the number of patch submissions from individuals and corporations was growing as well. He suggested that the current methods of breaking kernel development down into maintainers and sub-maintainers might not be able to deal with the massive, increasing, unholy deluge of patches coming in from all over the world. He said he'd already delegated a lot of his own work as maintainer to other people, but there just weren't enough people he trusted with some of the more intricate and subtle parts of the kernel to continue simply delegating things out.

Greg Kroah-Hartman replied in all seriousness that he thought, "I've been wondering if someone is trying to flood us with crap submissions just to try to DoS us and slow us down. If not, it's an interesting 'attack' vector onto our development process that we need to be able to handle better."

Thomas said he didn't think it was a denial-of-service attack; and Alan Cox remarked:

There are a small number of very large businesses that generate a lot of the money in the server space. They have individual needs and the money tends to drive chunks of kernel development. That's both a bad and a good. The bad is they are trying to get it to do what they need, now and without planning for the long term properly. The good is that they are paying developers who otherwise might be working on other stuff.

Thomas also added that a lot of businesses that paid employees to contribute to the kernel and assessed the performance of those employees by how many lines of code they got into the kernel or how many patches they had accepted.

ZACK BROWN
The Linux kernel mailing list comprises the core of Linux development activities. Traffic volumes are immense, often reaching 10,000 messages in a week, and keeping up to date with the entire scope of development is a virtually impossible task for one person. One of the few brave souls to take on this task is **Zack Brown**.

So, in fact, there could be an unintentional DoS going on, in which corporate performance evaluations result in employees submitting patches in such a way as to increase massively the number of individual patches, as well as lines of code, that are accepted into the kernel.

Troubled Maintainerships

Thomas Gleixner posed another question: "How do we handle 'established maintainers' who are mainly interested in their own personal agenda and ignoring justified criticism just because they can?"

Greg Kroah-Hartman replied: "The wonderful, 'how do we remove a maintainer who isn't working out' problem. It's a tough one, I don't think we really have any standard way. Luckily in the past, the insane ones went away on their own."

Alan Cox defended those maintainers, though, saying that in most cases they didn't have an "agenda" so much as they just thought their approach was the right way to do things, and that was what a maintainer was supposed to do anyway. He used the example of the highly contentious kernel scheduler, saying, "I'm firmly of the opinion that 99% of users would be better off if we took all the current scheduler nonsense and replaced it with a lightly tweaked version of Ingo's original O(1) scheduler."

But Alan did acknowledge that some maintainers had other things in their lives, such as new children and professional emergencies, that they considered more important than their work on the Linux kernel. He suggested that having two co-maintainers for a given portion of the kernel might solve those sorts of problems.

Mark Brown and Trond Myklebust liked that idea, and Trond suggested "demanding that all patches that are submitted to the maintainer contain at least one 'Reviewed-by:' tag."

Dave Chinner liked this idea and suggested that it could create a culture of patch reviewers, who would each give prompt reviews of other people's patches, so that those people would give prompt reviews of theirs and the maintainers would also learn which reviewers could be trusted to make good review decisions, thus reducing the overall load on the maintainer.

At a certain point, Greg started a new thread, in which he made certain promises as a maintainer (modulo catching the flu, or something like that):

Here's what I, as a kernel subsystem maintainer, will strive to do for all patches sent to me by developers:

1) I will review your patch within 1-2 weeks (see details below).

2) I will offer semi-constructive criticism of your patches.

3) I will let you know the status of your patch if it is rejected, or if it is accepted, what tree it has gone into, where you can find it, and when you can expect to see it merged into Linus's tree.

Also, consider the kernel merge window requirements; during the merge window, I can't accept new patches into my tree that are not bugfixes for this specific kernel release, so that means there is usually a 3 week window starting about 1 week before Linus's release, lasting until after a -rc1 kernel is released, before I can get to your patch. During that time period hundreds of patches accrue, so please give me some time to dig out from all of them. Usually by -rc3 I'm caught up, if not, email me and ask.

There was no direct response to Greg's set of self-guidelines, but it does seem clear that between patch submission overflow, and maintainership issues, there will probably be some new developments regarding the roles of maintainers, and the roles of patch submitters and reviewers in the larger contributor pool. ■■■