Zack's Kernel News

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

By Zack Brown

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**.

New PWM Subsystem

The new PWM subsystem almost got derailed when Thierry Reding submitted it to Linus Torvalds to be included in the main kernel tree.

The PWM (pulse width modulation) subsystem specializes in changing the widths of rectangular pulse waves, as a way of controlling the power of the signal. It's useful in minor, unimportant ways like controlling the fans that prevent your computer from melting into sludge and backlights that let you see what you're working on.

Thierry's work is intended to replace the old overly constrained API with a fancy new generic subsystem that can support many drivers. As he pointed out in his request, the code had been under development for a year and a half and had been in the *linux-next* tree for a couple of weeks without revealing any major issues. Many drivers had already been ported from the old API to the new subsystem, and the next logical step, Thierry said, was to incorporate the code into the new tree.

Linus had no objection to the code, but he had significant objections to the process Thierry had taken to submit the work. First, Linus wanted *Signed-off-by* tags from people who expected to be actual users of the PWM subsystem.

But even more important, Thierry had not been incorporated into the GPG web of trust. The web of trust dates back to the cyber assault against the kernel.org servers back in August 2011. The attack resulted in a long-term disruption of kernel.org services, some of which never came back or could only be restored in restricted ways.

One of the responses to the attack was the creation of a kernel developer web of trust. Key signing events occurred all over the world, where developers met face to face to verify each other's identities and digitally sign their GPG keys. Thierry lived a somewhat quiet, isolated life in Hamburg, Germany, and didn't know any kernel developers near him who might sign his key.

That all changed as soon as this problem came up. For starters, a bunch of PWM subsystem users all piled into the discussion, offering their "Signed-off-by" tags and affirming that Thierry's code was useful and good. On top of which, Sebastian Andrzej Siewior signed his GPG key.

Linus actually hadn't waited for the key signing, though. So many people had signed off on the code that it hadn't seemed necessary to wait. People like Arnd Bergmann, who had said, "Very much Ack on the new subsystem. It uses the interface declarations as the previously separate pwm drivers, so nothing changes for now in the drivers using it, although it enables us to change those more easily in the future if we want to. This work is also one of the missing pieces that are required to eventually build ARM kernels for multiple platforms, which is currently prohibited (among other things) by the fact that you cannot have more than one driver exporting the pwm functions."

Throwaway Device IDs

Jean Delvare tried to finagle a solution to a problem he'd seen with device IDs. Device IDs are useful for identifying any sort of piece of hardware you might want to mount as a block or character device. Without an ID, you wouldn't be able to refer to the device to mount it.

In some cases, no device ID is needed because only one device of a particular type can exist at a time. In that case, multiple devices of that type may be on the system, but only one would have an ID. All's well.

The problem Jean identified was when multiple instances of a single device type could exist, but where the ID didn't matter and couldn't be known in advance anyway. In that case, he wanted to implement some throwaway device IDs that could be given to these devices as needed.

His idea was that these throwaway IDs would all be given negative values. This would let the kernel know they were throwaways that could be ditched afterwards, and to the user, they'd just show up as normal IDs with a clarifying suffix attached.

The problem with this concept, as Greg Kroah-Hartman put it, is that it would "overload" the device IDs. Instead of being the simple designators of the devices on a given system, they'd now be a data structure whose elements had more significance than just to identify something by number. Greg suggested that instead of assigning an unusual meaning to negative numbers, Jean should simply add an additional boolean value for each device

whose number was a throwaway. The boolean wouldn't be too much of a memory drain, he said, and it would be preferable to saving a few bytes at the cost of clarity and simplicity.

This made sense to Jean, he reimplemented the device ID feature, and Greg accepted it into his patch, heading for Linus Torvalds's tree. Presto.

Optimizing Tracepoints

Steven Rostedt posted a very nice explanation of how kernel tracepoints are converted from source code to machine code [1].

His point was that after a number of interesting steps, the tracepoint would be converted into quite a lot of machine code, sitting right in the "hot path" (i.e., in parts of the kernel that are executed a lot). The problem with this is that a lot of the calculations done by the tracepoint code could be done someplace other than the hot path, where it slows the whole kernel down.

The actual amount of slowdown per tracepoint, he acknowledged, was microscopic, but with hundreds of tracepoints in the kernel, and more on the way, he was concerned there could very well be a real speed hit occurring.

Steven figured out a fix that could be implemented at the level of the kernel source, just by making a slight change to the way tracepoints are called initially. His question was: Should he make the change for all the tracepoints used in the kernel?

Mathieu Desnoyers was slightly skeptical that the small amount of machine code being generated on the hot path would be enough to cause a noticeable slowdown, but he suggested that even if it were, the best solution might be to approach the GCC maintainers and ask them to have the compiler detect this kind of code and move it out of the way automatically.

Steven replied that getting the GCC developers to listen to kernel requests would be about as likely as himself winning an Olympic gold medal.

In fact, in the old days, there were indeed epic battles between the GCC folks and the Linux kernel folks. At one point, the kernel folks were actually boycotting all recent (at the time) versions of the GNU C Compiler on the grounds that only one particular older version of the compiler could produce good machine code, whereas the GCC people were equally adamant that the kernel sources just needed to be fixed to make the best use of the latest compiler versions.

Those were scary times. Two immensely important and essentially interdependent projects, both central to the entire free software world, were filled with bitter resentment toward one another.

Back here in 2012, though, H. Peter Anvin replied to Steven, saying that those issues might have existed "at one point, maybe, but lately we have had a lot more traction from the gcc developers, giving us features like __fentry__ and asm goto."

At this point, the discussion changed to what sort of changes GCC might make to accommodate this particular problem; Steven seemed to feel it would not be a difficult feature to add, so assuming the slowdown is even significant enough to fix, it might not result in any changes within the kernel code itself, but to GCC instead.

Linux 3.2 Going Stable

Ben Hutchings announced his intention to maintain the 3.2.y kernel as a "long-term stable" tree for as long as the Debian folks supported Debian 7.0. He estimated this would take that kernel all the way through 2015, at which point, he said he'd be happy to hand maintainership of that kernel over to someone else.

The significance of long-term stable kernel trees is that Linus Torvalds no longer maintains or even initiates any sort of stable branch. In the old days, stability figured into at least some of his own work, but nowadays, the official kernel tree remains under very active development all year round. Other developers, like Ben, simply take over maintenance of a particular kernel based on their own desire to do so and bring it to a more and more stable state. In this way, their particular need – in this case, Debian 7.0 – can be served. Of course, just because Ben is doing this for Debian, doesn't mean the rest of us can't enjoy the fruits of his labors by using his stable tree for our personal systems or for the other distributions we maintain. ■■■

INFO

[1] Tracepoints: *https://lkml.org/lkml/2012/8/9/521*