Zack's Kernel News

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

By Zack Brown

Don't Trust Firmware

Over the course of debugging something, Marcin Slusarz remarked that the problem looked like it might be an ACPI bug, to which Linus Torvalds replied:

Here's the #1 thing to keep in mind about firmware:

*- firmware is *always* buggy.*

It's that simple. Don't expect anything else. Firmware is written by people who have lost the will to live (why? Because they do firmware development for a living), and the only thing keeping them going is the drugs. And they're not the 'fun' kind of drugs. The end result is predictable. In their drug-induced haze, they make a mess.

So saying 'ACPI is buggy' is like saying 'water is wet'. Deal with it.

He added:

You need to understand that ACPI has been tested with one thing, and one thing only: Windows. Clearly windows doesn't ask for some three-byte region. So it doesn't work. Big surprise. Untested code written by monkeys on crack – what did you expect?

So don't do 'clever' things. When it comes to firmware, you need to expect it to be buggy, and try to access it the way Windows accesses it.

There actually turned out to be some question about whether the problem really was with the firmware, or if it was actually with the kernel's ACPI implementation. The issue wasn't resolved on the kernel mailing list, but Linus's rant still seems like it might be useful to remember.

Patch Acknowledgement Process

In the course of hacking on Linux, Al Viro suggested a change to the patch submission procedures. He wanted there to be a *detached Acked-by object* that could amend the commit message of a given patch in a Git tree. He said, "The situation when ACKs come only after the commit has been pushed is quite common. Linus, what do you think about [the] usefulness of such [a] thing? Ability to append ACKed-by/Tested-by of an earlier commit to a branch instead of git commit --amend + possibly some cherry-picks + force-push, that is."

Thomas Gleixner agreed that this was a common situation, although he felt that, instead of simply modifying the commit, it would be better to keep a record of the belated Ack in the Git tree itself. He suggested that some semantics like

```
git --attach SHA1 <comment>
```

would be good.

Jeff King pointed out that

```
git notes add -m <comment> SHA1
```

was already available. He explained, "The resulting notes are stored in a separate revision-controlled branch and can be pushed and pulled like regular refs. Note, though, that the default refspecs do not yet include refs/notes, so you'd have to add them manually. The workflows around notes are not very mature yet, so if you start using them, feedback would be appreciated."

Thomas was very enthusiastic about this feature, and he and several others started peppering Jeff with feature suggestions. But regarding this Git feature, Linus Torvalds remarked, "Don't use them for anything global. Use them for local codeflow, but don't expect them to be distributed. It's a separate 'flow', and while it *can* be distributed, it's not going to be for the kernel, for example. So no, don't start using this to ack things, because the acks *will* get lost."

However, Al did want something that could be used for the official kernel tree. He suggested a

```
git commit --allow-empty
```

where the comment field could contain a belated Ack for a specific commit, although he wasn't sure it was in good taste to allow something like that. Linus replied, "Don't bother. It's not that important, and it's just distracting. It's not like this is vital information. If you pushed it out without the ack, it's out without the ack. Big deal."

Ingo Molnár also pointed out that patch Acks were mostly useful before the patch was accepted, as a way of making sure that at least someone else's eyes had been on it before it went into the kernel. Once in the kernel, Ingo said, Acks were mainly only useful for bureaucratic purposes.

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**.

Dissatisfaction with udev Maintainership

Mauro Carvalho Chehab complained that the udev developers had not addressed some very important and longstanding breakages. He summarized a variety of things that he and others had done to try to fix or work around the problems, and concluded:

For Kernel 3.6, we'll likely apply some quick hack. However, for 3.7 or 3.8, I think that better is to revert changeset 177bc7dade38b5 and to stop with udev's insanity of requiring asynchronous firmware load during device driver initialization. If udev's developers are not willing to do that, we'll likely need to add something at the drivers core to trick udev for it to think that the modules got probed before the probe actually happens.

Linus Torvalds promptly blew his stack at the udev maintainers, saying, "udev made new – and insane – rules that are simply *invalid*. Modern udev is broken, and needs to be fixed." He called upon Greg Kroah-Hartman, one of the original udev authors, to step in and make sure a good fix was found.

Greg did some debugging to try to find out exactly when udev started having problems, and Linus rooted around in the Git history for that answer. He didn't like Mauro's idea of implementing a workaround in kernel space. He said, "I think it would be absolutely insane for the kernel to work around the fact that udev is buggy."

Kay Sievers, one of the udev maintainers, responded to folks' comments. He said that udev wasn't broken, it just had a delay that eventually would time out; it was this timeout that caused problems. He said that drivers were blocking somewhere in user space; Kay felt it was up to the kernel to handle that because the drivers had no business depending on user space at all.

Linus didn't like that explanation whatsoever. He didn't think the kernel was doing anything wrong, especially since the whole situation had been working fine before certain patches went into udev. In spite of his earlier wish to avoid doing a workaround that avoided udev, he did post an aggressive workaround.

Kay reiterated that it was the kernel itself that was causing the breakage and that udev's only problem was that it was slow. He said he and the other udev developers were working on fixing that, but that they hadn't wrapped their heads around the real cause of the slowdown yet; however, he said Linus's patch seemed like a good approach.

Linus blew his stack at that all over again, saying:

*Yes, doing it in the kernel is 'more robust'. But don't play games, and stop the lying. It's more robust because we have maintainers that care, and because we know that regressions are not something we can play fast and loose with. If something breaks, and we don't know what the right fix for that breakage is, we *revert* the thing that broke. So yes, we're clearly better off doing it in the kernel. Not because firmware loading cannot be done in user space. But simply because udev maintenance since Greg gave it up has gone downhill.*

Alexander Viro said to Kay, "Just fix udev, and if you can't fix it someone please just fork the last working one."

Module Signing Simplification and Motivation

Linus Torvalds sometimes proposes kernel features in public. You might think he'd just implement the things he wanted, and sometimes he does, but when he's working in a part of the kernel that he's not expert in, or when he's proposing radical changes, he'll often go through the normal chain of command as if he were an ordinary contributor, sending patches to official maintainers of particular features and whatnot.

This time he was working in the area of module signing, which allows the kernel to verify that a given module has been approved by some kind of authority, such as a machine owner, a distribution maintainer, or anyone else who might be putting a Linux system together. He posted his suggestion as a public RFC because it represented a radical change to the code. Apparently, David Miller had complained that module signing was dramatically slowing down the build process. Linus's idea was to move the key signing from build time to install time, where a much faster process could accomplish the same thing.

David Howells was enthusiastic – perhaps too enthusiastic – about Linus's patch. He suggested simplifying the code even further by eliminating automatic module signing entirely and letting signing be done entirely by hand instead. Linus replied, saying "That's just disgusting crazy talk. Christ, David, get a grip on yourself. You seem to dismiss the 'people want to build their own kernel' people entirely."

Linus said that one of the best use cases for module signing involved people who wanted to generate a random key,

use it to build a one-time kernel with associated modules, and then throw away the key files. Those users would then have a system that would be much less vulnerable to rootkit attacks.

He said that an alternative use case, in which a distribution used the powers of key-signing to control a user's system against their will, "is garbage and should largely be ignored."

Linus went on, "In contrast, the case I outlined above is about true *security*, not some 'vendor control' bullshit. And THAT is the case that kernel developers should encourage: using cryptography to secure individual users, instead of controlling things for others."

He added, "your 'get rid of all the automatic module signing stuff completely' answer makes me think that your agenda is fundamentally flawed," and he went on, "if I believed this was about redhat or microsoft keys, I would never have merged the code at all."

The rest of the thread was mostly a technical discussion of Linus's original patch. Rusty Russell posted a new patch that fixed some things that Linus had pruned too aggressively, and Josh Boyer also posted some improvements as well. Linus liked these changes, and the three of them – and a couple other folks – continued developing the patch for awhile.

Tracking Boot Speed

Lee Jones sent in a patch by Jonas Aaberg that tracked how much time a system took to boot up, along with other relevant data, such as what kind of load the system experienced during the boot. The idea was that this kind of information would be useful for debugging and optimizing the boot code.

Christian Gmeiner pointed out that the code wouldn't work under x86, and Lee promptly invited him to extend the patch to support that architecture.

Nishanth Menon offered some bug reports, and Lee promised to submit some updated fixes. Shortly thereafter, Lee posted a big update, and Dan Murphy reported that the code compiled fine but reported identical bootup times for two independent systems – a highly improbable case. They proceeded to debug the issue.

Russell King also looked at the patch and was concerned that some of the files the code created in SysFS had ProcFS ugliness, such as special formatting and multiple pieces of information per file. But, Arnd Bergmann pointed out that, although the files were located in the `/sys` directory tree, they actually used DebugFS as their underlying filesystem, which had no such ugliness restrictions.

The whole email thread was essentially about ways to enhance the patch and correct any problems, so it seems as if there's plenty of interest in this sort of information going into the kernel. As Lee said in his original email, the overhead for the kernel is very low; so, I expect this patch to get into the kernel at some point in the not too distant future.

Power Tracking

Pawel Moll had been maintaining some power usage graphs for ARM, his employer, and they started asking for information that wasn't yet provided by the kernel. They wanted `ftrace` and `perf` to output power usage information as well, and they also wanted a CPU frequency governor that could adjust the speed of the CPU in response to power utilization.

Pawel readily admitted he didn't know how to go about creating a frequency governor (although he had an idea or two), but he had come up with three possibilities for the ftrace/perf issue.

His first idea was to make power usage data available to ftrace and perf by defining a simple trace event that would allow any sort of monitoring driver, such as hwmon, to track the relevant data right away. He suggested a couple of possible trace event definitions that might work.

However, a trace event would not produce data that the user could use as a normal data source, so it wasn't an ideal solution. Pawel's second idea involved implementing a perf Performance Monitoring Unit (PMU) that would treat power usage as environmental data and make it available through the SysFS interface. Or, a separate driver could register its own PMU without requiring any in-kernel code and likewise export the data to SysFS.

His third idea was to create a dedicated kernel API that would allow power data to be collected and distributed to whichever user needed them. This would also help, he thought, with designing a CPU frequency governor.

Steven Rostedt had a few comments, but mainly he pointed out that any in-kernel solution would have to take account of really big systems, with massive numbers of CPUs working simultaneously. This was not the use case Pawel had considered, and it put some new constraints on any solution.

Andy Green also suggested that even a single-CPU system could have additional relevant power consumption data, such as the effect of DDR RAM on power consumption. He suggested that any solution should incorporate the ability to track other types of power usage on a given system.

Guenter Roeck also replied to Pawel's third idea, saying that an in-kernel API had been tried a couple of times and that "the result was, at least so far, more complexity on the driver side. So the difficulty is really to define an API which is really simple, and does not just complicate driver development for a (presumably) rare use case."

A few other issues cropped up, and various suggestions were made. Thomas Renninger suggested that the whole thing might just be best solved in user space; indeed, that seemed to be the direction Pawel was heading, at least for the moment. Typically, the existence of a userspace solution to a given problem means it probably won't be addressed inside the kernel, so the most likely scenario for Pawel's (and ARM's) problem seems to be that he'll just have to keep looking for userspace solutions.

Empty Files in the Kernel Tree

David Howells noticed an empty file in the kernel source tree, and Linus Torvalds replied, "Sadly, this is by far not the only empty file we now have."

Linus said, "Doing a 'git clean -dqfx' followed by 'find . -size 0' shows quite a few empty files, most of them being Kbuild files, and most of them having come in through the UAPI patches. Maybe some of those *should* be removed?"

He increased the size of the CC list for the mailing list discussion, and Viresh Kumar claimed responsibility. He was very surprised that the empty files had been added by his commit, and posted a patch to get rid of them.

Arnd Bergmann accepted the fixes and promised to push them upstream with his next round of submissions. ■■■