

ZACK'S KERNEL NEWS

Status of HGA Framebuffer Driver

The HGA Framebuffer driver is no longer maintained. Roland Kletzing confirmed that the official maintainer, Ferenc Bakonyi, has not had the relevant hardware since 2001. Thus, the MAINTAINERS entry is completely out of date and Roland has posted a patch to remove it.

Fourth-Party Drivers?

Michael Buesch has been designing and building his own hardware devices and recently asked whether he should submit drivers for that hardware or just keep them as a separate patch. The overwhelming response was, "Send in the drivers!" Apparently, as long as Michael – or anyone else – is willing to document how folks can build the hardware for themselves, the drivers for that hardware will be a welcome addition to the kernel. With that kind of encouragement, hopefully we'll start seeing a variety of unique hardware drivers appear.

The Linux kernel mailing list comprises the core of Linux development activities. Traffic volumes are immense, often reaching ten thousand messages in a given 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. Our regular monthly column keeps you abreast of the latest discussions and decisions, selected and summarized by Zack. Zack has been publishing a weekly online digest, the Kernel Traffic newsletter for over five years now. Even reading Kernel Traffic alone can be a time consuming task. Linux Magazine now provides you with the quintessence of Linux Kernel activities, straight from the horse's mouth.



Removing Firmware from the Kernel Tree

David Woodhouse wants to remove all third-party firmware out of the kernel source tree. David is in favor of letting the kernel load arbitrary blobs of firmware code (provided the resulting binary can be distributed legally under its license), but he doesn't think such things belong in the kernel. He's working on creating a separate git tree for all of them.

A number of replies dissented from this. Most folks were in favor of isolating firmware into a single location, but taking it out of the tree seemed too much because it created issues of competing firmware packages and other requirements that some folks don't want to tackle.

The most vocal opponent of David's idea was David S. Miller, who has been fighting against this idea for a while, specifically regarding the tg3 driver, which he felt needed to keep its firmware in the main tree. David Miller said that if distributions like Debian wanted to avoid any binary-only data in their kernel, it was up to them to write and maintain the necessary patches out-of-tree.

David Woodhouse responded that he'd already taken steps to ensure that even without removing the firmware, his changes would still be useful, but he did feel it would be better to take them out entirely. At this point, folks turned the discussions into a technical consideration about how the idea might be implemented. Whether some of the ideas were intended as actual suggestions or just as ruminations on what might be possible if such a thing were desired was unclear. What was clear is that a lot of folks had a variety of opinions, with many different reasons for them. In terms of what might happen with David Woodhouse's patches, I'd expect some kind of compromise. Removing the firmware is something that free software purists prefer, but it also might be something preferable for practical, legal reasons. In that case, folks like David Miller could lose the battle, but hopefully without too much inconvenience being forced on them.

Helping Automate Kernel Builds

Clifford Wolf has written a new makefile target, no2modconfig. Ordinarily, when you configure your kernel for compilation, any option in which you type *N* will not be compiled. With no2modconfig, those items are compiled into modules. Apparently, this is useful for auto-generating kernels.

Sam Ravnborg, it turns out, has some patches that do what Clifford wants in a way that uses the existing allmodconfig build target with a predetermined base config file. In response to Clifford's posting, Sam said he'd try to get his own solution ready to be reviewed in the near future.

Writing Under CramFS, SquashFS, and Others

Arnd Bergmann added write support to CramFS ... sort of. Now the user is able to modify files on a running system, but the files themselves are never written to disk – the changes just hover in memory, and a reboot returns the system to its original state. Phillip Lougher says he's also planning to implement this sort of thing for SquashFS. Typically, past attempts involved the use of UnionFS to overlay a TmpFS instance on the filesystem in question. Arnd pointed out that UnionFS isn't actually the best solution and that coding the feature directly into CramFS is preferable to the hackiness of UnionFS. Phillip added that stackable filesystem support really belongs in VFS, which doesn't currently support it.

Enough people advocated using – or at least fixing – UnionFS to create a technical discussion about how that might be accomplished. Arnd and Phillip both said they'd be happy to compromise if UnionFS (or AUFS) could be made to do what they needed without too massive an effort, but the discussion ended inconclusively. To the kernel folks, it seems clear that pseudo-write support on CramFS and SquashFS (and possibly ISO 9660 as well) would help users, and the features should be supported one way or another.

Direction for 2.4

After what appeared to be a long hiatus, Willy Tarreau has been releasing more versions of the 2.4 kernel lately. Also, he's made some effort to compile some statistics about who uses that kernel and why they don't upgrade to the 2.6 kernel. With only 22 respondents, Willy's numbers have a pretty big margin of error, but they're still interesting. About half of the folks said they relied on 2.4 for general-purpose servers on which outages and upgrade regression problems would inconvenience a number of people, so they don't bother to upgrade to 2.6 and tend to keep the latest 2.4. About 20 percent of the respondents use 2.4 for application-specific servers, which tend to be "mission critical" servers on which any outage – even upgrading to the most recent 2.4 kernel – would be a big problem.

Approximately 10 percent of respon-

dents use 2.4 for routers, firewalls, and other network deployment applications. In those cases, upgrading to 2.6 could be relatively easy because of the few system requirements, but folks aren't sure how to upgrade successfully, so they just stick with what they know.

Another 10 percent of respondents use 2.4 in their embedded systems products. In this case, taking the systems down for an upgrade would be visible to the customer and require substantial changes to the company's whole build process.

The rest of the respondents, approximately another 10 percent, run 2.4 on their personal systems, either because they didn't want to try to configure a new kernel for that hardware or because they were used to working on the system as-is and don't want any change at all.

Willy conducted the survey in part to figure out how to encourage more people to upgrade away from 2.4 so that tree could fade away peacefully. Willy's main recommendation for this was that the 2.6 folks produce a clear description of the functional differences between 2.4 and 2.6, so users are not left trying to figure out what is going to break on their own, and also that it is easily possible to go back to 2.4 if the upgrade doesn't go well.

Another reason for the survey was to make adjustments to Willy's 2.4 release schedule. He said, "I will issue stable releases a bit more often for users to quickly get their fixes, but progressively increase the delay between major releases." He continued by saying that major releases would only be issued with new PCI IDs, major driver updates, and compiler support, for example.

