Zack's Kernel News

INFO

The Kernel Mailing List comprises the core of Linux development activities. Traffic volumes are immense 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 that take on this impossible task is Zack Brown.

Our regular monthly column keeps you up to date on the latest discussions and decisions, selected and summarized by Zack. Zack has been publishing a weekly digest, the Kernel Traffic Mailing List for several years now, reading just the digest is a time consuming task.

Linux Magazine now provides you with the quintessence of Linux Kernel activities straight from the horse's mouth.

Rootkit setback

Rootkit authors have been dealt a temporary setback with the 2.6 kernel, as that kernel series no longer exports the system call table. Many rootkits in earlier days would use this table to intercept system calls, and thus gain root that way. Anyone thinking that Linux has the rootkit authors beat, however, should think again.

There are still quite a few ways to go about trying to gain root on a Linux box. Some still involve intercepting system calls, just without using an exported system call table. In that scenario, the rootkit looks for any exported symbol, and then looks around in the memory near that symbol until it finds something that it can use.

Existing rootkits such as the "adore-ng" rootkit have already been ported to Linux 2.6, with all the same features still offered. In the case of "adore-ng", the rootkit author had abandoned the system call table as the point of entry, and instead planned to gain access to vulnerable systems via the Virtual Filesystem layer. ■

A sound idea

Often, kernel developers will try something really geeky like transferring generic data through a sound card. The best part is that, as often as not, they find a way to do it! That's exactly what happened this past February. Nischal Saxena, Jamie Lokier, Pavel Machek and others started knocking the idea around, and Pavel was actually able to transfer data from the beeper on his PC, to the internal microphone of his notebook.

Paulo Marques estimated that SPDIF might give up to 4.6 megabits of data transfer per second; and that theoretically, computers linked by their sound card inputs and outputs could be mapped as regular nodes in a network. So far there doesn't seem to be any actual effort to get something like this into the kernel itself, but stranger things have happened, such as displaying the output of kernel panics in morse code. ■

Your style or mine

Over the years, the CodingStyle document distributed with the kernel sources has served as a loose guideline for how kernel coding should be organized and formatted. Not everyone adheres to these guidelines, in fact many developers have strenuously objected to stylistic principles that violate their own ideas of best practices. Nevertheless, periodic "clean-up" patches do appear on the linux-kernel mailing list, to bring various portions of the kernel into compliance with the current state of the CodingStyle document.

One recent attempt to bring CodingStyle up to date with various preferred practices, included allowing developers to skip the apostrophe in "don't" and "can't" in their code comments, making them "dont" and "cant" instead.

This particular change does have a little support among kernel developers, one reason being that the apostrophe character is often used as a delimiter, and so can cause confusion during parsing. Fortunately, good English prevailed and the CodingStyle document was instead updated to admonish developers to either use apostrophes or avoid contractions.

It was also suggested that the 80-character limit for line-length within the kernel sources was far too restrictive, as many developers these days used xterms with far more than 80 columns. Andrew Morton stepped into this debate, clamping down on any effort to expand the limit.

At one point he said, "Yes, 80 columns sucks and the world would be a better place had CodingStyle mandated 96 columns five years ago. But it didn't happen." As it turns out, there are still developers – notably David Weinehall, the 2.0 kernel maintainer – who still use screens that are limited to only having 80 columns in width. So although the 80-column limit may one day go away, it seems clear that it will not start in the 2.6 tree. ■

Open and Closed

Intel has begun a free software project to support the Intel PRO/Wireless 2100 miniPCI network adapter. The company-has initiated a SourceForge project, along with a mailing list, and implemented support for kernels in the 2.4 and 2.6 series.

In keeping with the open source tradition, Intel had released the code in an early BETA form, to encourage development and bug reports from the community in an ongoing way. However, there does happen to be some closed source firmware that goes along with the driver. James Ketrenos, the project leader, has said that this firmware is loaded and executed entirely within the hardware, and at no time operates within the Linux kernel, or knows anything about it.

The issue of closed source firmware has always been a sore point with Linux and the free OS community in general; but there do seem to be compromises taking place on both sides, as this Intel project would indicate. Intel itself is willing to develop some open source drivers in a friendly way that benefits all, but there are still some hardware details that Intel regards as secret, as they give Intel a competitive lead that the company wants to protect.

Hopefully these details will gradually come out as well. ■

Debugging a debugger

A large-scale, concerted effort appears to be underway, to get the KGDB kernel debugger into the 2.6 tree. Linus Torvalds has traditionally opposed any such idea, claiming that it allowed developers to get lazy with debugging; but other developers are not so strongly against it, in particular Andrew Morton, the official 2.6 maintainer. So it looks as though KGDB will finally make it into the kernel after all.

One technical problem has been a proliferation of independent KGDB-related patches, some of which conflicted, and some of which were merely redundant, making it difficult for the project to come up with a coherent set of submissions. Starting in late January, Tom Rini set up a BitKeeper repository for any KGDB work, in an attempt to merge everything together, and to discard the redundant bits.

A short time later, Amit S. Kale set up a similar project on SourceForge, replacing the BitKeeper tree with CVS, most likely because of BitKeeper's closed source status. Later in February, Tom and Amit were clearly working together with various other folks like Pavel Machek; with the kernel maintainer, Andrew Morton checking in to see if any of the work was ready to be merged into the 2.6 tree.

It was still a bit soon at that point, but by early March, Amit and the rest had organized the KGDB patches into several smaller groups and begun to talk about how best to submit the patches so that they would be acceptable. Some of the patches entered into a "feature freeze" at this time, while others continued to be developed.

The frozen set constituted of a "KGDB lite" patch set, containing what appeared to be the most essential aspects of KGDB. Other features, it was reasoned, would be grafted on later after KGDB-lite had been accepted. At around this time, Amit also published a batch of KGDB documentation, explaining how to compile and use a kernel with KGDB support. Finally, in mid-March, Amit submitted a set of patches that he and the other developers thought had the best chance of being accepted by Andrew. It turned out that Andrew was actually willing to take more than just the "lite" patches; in fact, he insisted on having features that the KGDB developers had thought would make the patch less palatable.

While Andrew was not fully aware of the ins and outs of the work going on in the KGDB code base, it came out that some of the features he wanted might make the code a little bit uglier, perhaps too ugly for acceptance. Work continued throughout March, with various nipping and tucking, trying to keep the code clean while still providing the most desired features.

Currently, work is still ongoing, but it seems clear that sooner or later, 2.6 will include a full blown KGDB implementation. ■