Zack's Kernel News

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

By Zack Brown

New Union Filesystem

Miklos Szeredi announced his own attempt at a union filesystem, in other words two distinct filesystems that are overlayed so as to appear as one. The two filesystems are ranked "upper" and "lower," so if a file in one filesystem has the same name as a file in the other, only the file in the upper filesystem will be displayed to the user. But, if a directory in one filesystem has the same name as a directory in the other, the two directories are presented as one. Only the files have the upper and lower selection method. Union filesystems are cool. Neil Brown explained these ideas and many other things in some documentation that he posted in response to Miklos's code. Miklos was overjoyed by this, in fact Neil's documentation immediately helped expose some bugs in Moklos's code, which he promised to fix right away.

After writing the docs, Neil's next step was to critique the code itself, and he and Miklos had a lively debate on a range of technical issues, essentially ironing out why certain things wouldn't work and why other things were easier than they'd been supposed. And, Al Viro and a bunch of others got into the mix, and the discussion ranged all over the place.

When Kernels Are Released

Piotr Hosowicz wrote a script to let him know when a new kernel version had been uploaded. But the script queried the kernel.org servers, which caused problems, because sometimes he'd get the master server, and sometimes he'd get the clones. The result was that sometimes his script would report a new kernel version when there wasn't one. He asked for help identifying the bona fide master server; but Jiri Kosina replied that Piotr wasn't using the best approach. Jiri recommended running `finger @kernel.org` to get the most up-to-date information. But Arnd Hannemann pointed out that this, too, was not the correct approach. Apparently, the finger daemon had been getting so many hits that, even in 2003, it was already overwhelmed and was crashing much of the time. Randy Dunlap offered a link to his own script [1] that would read the finger banner data from *http://www.xenotime.net/linux/scripts/kcurrent*?? and report the changes. Apparently using Randy's script, or just querying the finger banner URL directly, is the real way to go.

New dd Clone

Douglas Gilbert announced that he and Mark Knibbs and some others were releasing `ddpt`, a rewrite of the Unix `dd` tool for writing to disk. He described some of the differences between ddpt and the original dd tool [2]. In particular, it specializes in SCSI block devices and derives its name from the "pass through" interface on those devices, that gives users fine-grain control over the data copy.

Planning For Stable Trees

Geoffrey Said pointed out that, in his work, the development process had to pick a kernel to stabilize on long before the kernel developers picked a kernel to make stable. Specifically, he and his people had chosen the 2.6.34 kernel on which to base their project, with the idea that SUSE had also chosen that kernel so it was likely to be supported in the long-term as a stable tree. But, alas, he said, the 2.6.35 tree is now the one being maintained. He suggested that the kernel folks decide much earlier in the process which kernels they intended to maintain as a stable tree.

Américo Wang suggested that Geoffrey pick a kernel and then just back-port features from whatever the stable kernel series ends up being.

Greg Kroah-Hartman, one of the stable tree maintainers, said that there really wasn't a set procedure for picking which kernels were going to be supported long-term. He said that anyone interested could just ask, and maybe someone like Greg or one of the other developers would have a good answer. So, Geoffrey asked whether perhaps 2.6.36 would be maintained as a stable tree, and Greg replied, "Nope, I am planning on sticking with .32 for a while now. Of course, all of the kernel releases will get the normal 3-6 months of stable support like always."

Console Checker

Dr. Werner Fink coded up a feature to add a `/proc/tty/consoles` file, which would show the character devices being used by the system console. So, for example, `/dev/tty0` and `/dev/ttyS0` might be listed in `/proc/tty/consoles`. There wasn't much comment about the code, and no real dissent; so it's still a bit up in the air whether it will be incorporated into the kernel.

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**.

Kernel Podcast

Jon Masters announced the return of his Kernel Podcast project [3] with transcript [4]. His main motivation for doing this is to force himself to keep up with the unfathomably high volume of linux-kernel mailing list traffic. It's a daunting task, with a "high risk of burn-out," as he put it. He had opted to take a break rather than quit altogether and gave some stats on how many folks were downloading the new podcasts so far. Since May 2009, he's had a total of 200,000 downloads. Welcome back, Jon!

IRQ Maintainer

Once in a while, a new entry is added to the MAINTAINERS file, not because there's a corresponding new feature of the kernel, but because there's a very old part of the kernel that someone has just become the right person to patch. Recently, Joe Perches suggested that the IRQ subsystem be given its own entry in the MAINTAINERS file, listing Thomas Gleixner as the official maintainer. Joe's patch stats showed that Thomas submitted almost 50% of all patches to that subsystem, with the next most prolific contributor submitting only 13%. There was no discussion or dissent regarding the patch, and Joe's patches to the MAINTAINERS file seem to be accepted most of the time; so I'd expect Thomas to appear in there shortly.

High-Availability Virtual Machines

Fernando Luis Vazquez Cao and others are trying to enhance the robustness of KVM virtual machine code to improve its "high availability" integration, for greater uptime and reliability. Fernando posted a lengthy description of their approach and asked for guidance on their direction. Their current idea is to use existing high-availability techniques like polling, where a given virtual machine is simply checked regularly by a separate process. Fernando said they wanted to complement that approach with one that was more event-driven. If you rely only on polling, there's a period of time between each check, where you won't be informed of an outage. And, if you increase the frequency of polling checks, the polling process uses more system resources.

An event-driven approach means the virtual machine would trigger some kind of alert when it crashed. The benefits of that are that it happens instantly in the event of a crash and doesn't use significant resources when the virtual machine is running well. One reason why polling is used instead of an event-driven approach is that relying on the behavior of something that's in the process of crashing is inherently risky. It might crash in weird ways that also affect the ability to trigger the event. Fernando pointed out that, in the case of virtual machines, there's only so much damage a crash can do. So, the outer system will be safe enough from the crash, to reliably trigger the event.

Another element of Fernando's approach was the idea that a crashed virtual machine could be preserved for later forensic analysis. However, David Lang suggested that it wasn't necessary to write special crash-preservation code, because it might be identical to simply pausing the virtual machine, which could be done already.

David had other comments about Fernando's proposal. He suggested that it would still be important to include polling in cases where the problem was more subtle than just a system-wide crash. For example, David said that if a networking cable went bad and data wasn't going through the connection, it would be hard for an event-driven model to distinguish that from a case where there simply wasn't supposed to be any data transfer. Overall, however, there was no objection to the idea of an event-driven model; just to some of the implementation details. So, likely Fernando's code, or something like it, will be integrated into the KVM code relatively soon.

INFO

[1] Randy Dunlap's script: *http://www.xenotime.net/linux/scripts/kcurrent*
[2] The ddpt utility: *http://sg.danny.cz/sg/ddpt.html*
[3] Kernel Podcast project: *http://podcasts.jonmasters.org/kernel/kernel.xml*
[4] Kernel Podcast transcript: *http://www.kernelpodcast.org*