ZACK'S KERNEL NEWS

exFAT Support

Someone pointed out on the list that Microsoft's exFAT filesystem seemed to be their answer to large portable flash drives and asked what, if anything, was being done in the Linux world to support the exFAT filesystem. Hirofumi Ogawa replied that he'd already written a read-only driver, but because of time constraints, he had not been working actively on adding write support. He posted his code, and H. Peter Anvin asked whether there were any filesystem specs available that didn't require signing away the ability to write code to them. Hirofumi replied that his own work had been based on reverse-engineering the filesystem on disk. Meanwhile, Alex Buell converted Hirofumi's patches to a standard kernel module that can be compiled outside of the kernel, just as long as the system has a recent kernel installed. The code is available for download at: *http://www.munted.org.uk/programming/exfat.tar.bz2*.

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.

GPL, LGPL, and IBM IP

Mathieu Desnoyers wanted to release the userspace RCU code he worked on under the LGPL instead of the GPL so that proprietary code could link with it and use it. RCU, read-copy-update, is a library that ensures that data objects defined in the kernel do not appear undefined to other running code that tries to access them before the definition process has completed. Defining and initializing a *struct*, for example, could expose it in an incomplete form if the compiler or CPU tries to optimize the data assignments and puts the *struct* assignment itself ahead of the code assigning values to the variables within that *struct*. Mathieu's library makes protection against this available to user space. He wanted to know whether switching to the LGPL would be legal and acceptable to the Linux community.

Alan Cox pointed out that IBM owned the patent on the RCU idea and they had only released the patent for use in GPLed code, so Mathieu would need to get permission from them before proceeding.

Ktrace Sleeper Kernel Tracing

Jiaying Zhang created ktrace, a mechanism for tracing kernel events by inserting tracepoints in the kernel code. The reason for this innovation is that the existing markers code are deemed too heavy-weight. By simplifying the design, Jiaying and the other ktrace developers found significant performance enhancements. Some prototype code is posted.

Zedtrace Tracepoint Filter

Tom Zanussi posted his own kernel tracing tool, zedtrace, with a minimal home page at: *http://utt.sourceforge.net/zedtrace.html*. It lets the user trace any tracepoints already defined in the kernel binary. Sophisticated filtering is provided by the Perl language. Python and Ruby support, he said, was on the way. Christoph Hellwig said he liked Tom's tool but felt that tracepoints themselves were badly in need of revision. He acknowledged this wasn't zedtrace's fault.

Status of exofs

Boaz Harrosh submitted an update to the exofs filesystem. Exofs is designed for Object Storage Devices (OSDs), a relatively new type of storage device that takes object orientation into the drive itself. Instead of dealing with a character stream or with blocks of related data, OSDs present data as a collection of objects, that in turn may be composed of other objects. The idea is that this higher level view into data allows the user more control over managing the data and providing security.

The new version is substantially updated. It replaces the kernel-based mkexofs tool with a userspace library that exports the same API. It also incorporates fixes that have gone into ext2 in the recent past. Because exofs is a fork of the ext2 code base, any fix to ext2 is likely to be a fix to exofs as well. Boaz also modified the API used to perform object access. Exofs exports a standard block-oriented interface, so that exofs filesystems can be treated the same as any other block-based filesystem; it also provides an additional API to support OSD calls. The previous version used an API from IBM that was different from the standard OSD API. The new version uses the open-osd API directly.

VMUFAT Filesystem for the Dreamcast

Adrian McMenamin explained that the Sega Dreamcast visual memory unit implemented a filesystem similar to FAT16. He posted a driver for the VMUFAT filesystem (culled from an earlier attempt by him long ago) and said he intended to document the entire filesystem and write a userspace tool to create a VMUFAT image. He also pointed out in the KConfig description that this filesystem was really very specialized for the Dreamcast and would not make a great choice on other drives. A bunch of folks offered feedback, mainly pointing out a few legacy bugs left over from Adrian's earlier code. By the end of it, Adrian decided to do a more thorough rewrite and aim for inclusion in a later kernel.

Auto-Suspend

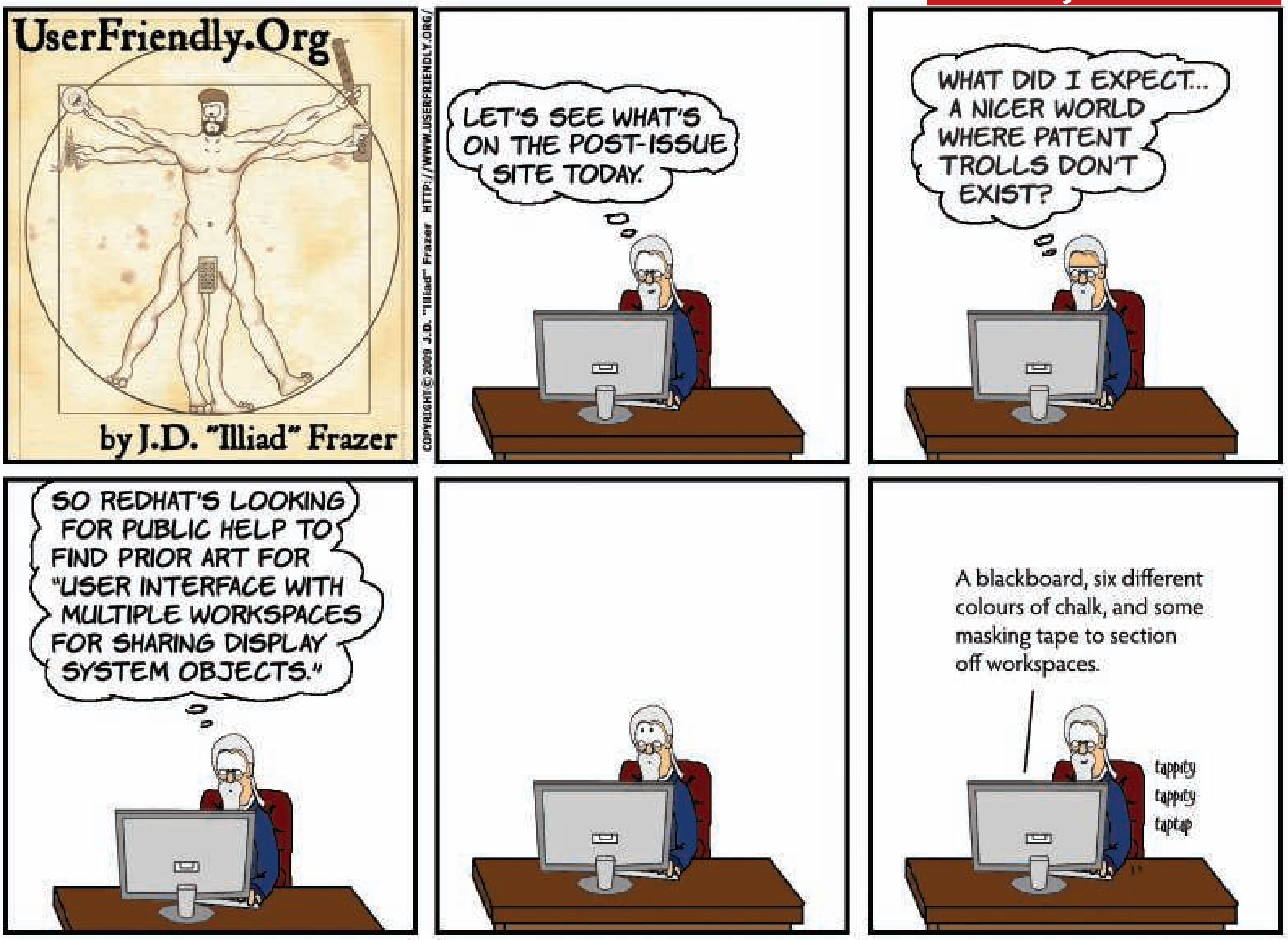
Rafael J. Wysocki pointed out that phones and other small devices necessitated the ability to suspend the system to RAM or to a low-power mode automatically after a certain period of idleness. He said that to implement these features in Linux, developers had to figure out what parts should be in the kernel and what parts in user-land. He enumerated the various questions, such as how to tell when the system was idle. In response, Arjan van de Ven and Benjamin Herrenschmidt opened up the debate, essentially agreeing that a device driver would have to make those kind of determinations. But as Roland Dreier pointed out, the situation is more complex in the case of highly integrated hardware because different parts of the system can be suspended at different times, and various parts of the system can be cleared to be suspended if various other parts of the system are already suspended. On a PC-like system, he said, a device driver would be sufficient to manage system suspension for the whole machine, but for highly integrated systems, many more options must be considered.

The discussion got very technical. The problem appears to be that the kernel shouldn't concern itself with the state of various user processes (i.e., userspace code should watch all that), whereas userspace code shouldn't concern itself with shutting down various internal parts of the system (i.e., the kernel should handle all that). This issue supplies plenty of examples, counter-examples, and special historical cases to back up everyone's point of view. Ultimately, a fascinating topological border will no doubt be established between the userside and kernel-side obligations. And whatever decision is ultimately implemented, it will undoubtedly set precedent for the next big question of where the kernel ends and the rest of the world begins.

To Dream the Unbreakable System

Tarkan Erimer had the idea of a "failover" kernel, in which two kernels would run simultaneously, with the primary kernel doing all the work and the failover kernel taking over in the event of a panic or other crash and simply continuing to run the system without interruption. Willy Tarreau pointed out that scheduling both kernels could be a thorny problem. Also, he explained that most system crashes were the result of non-recoverable errors, like memory corruption or a driver bug. In those cases, a failover kernel wouldn't be able to sanely recover the system because many conditions would require a reset to restore to a known state. Memory corruption could end up worse than before. He also mentioned that he has implemented a similar concept with the use of a watchdog timer that would reboot to a second kernel image if the system crashed for any reason.

Linux Magazine Exclusive


UserFriendly.Org
by J.D. "Illiad" Frazer
COPYRIGHT © 2009 J.D. "Illiad" Frazer HTTP://WWW.USERFRIENDLY.ORG/
LET'S SEE WHAT'S ON THE POST-ISSUE SITE TODAY.
WHAT DID I EXPECT... A NICER WORLD WHERE PATENT TROLLS DON'T EXIST?
SO REDHAT'S LOOKING FOR PUBLIC HELP TO FIND PRIOR ART FOR "USER INTERFACE WITH MULTIPLE WORKSPACES FOR SHARING DISPLAY SYSTEM OBJECTS."
A blackboard, six different colours of chalk, and some masking tape to section off workspaces.
tappity tappity taptap