ZACK'S KERNEL NEWS

Hard Core Filesystem Tests

Eric Sandeen decided to use Steve Grubb's fsfuzzer tool to randomly alter bits on an ext3 filesystem. his plan was to alter the bits and then test to see how well ext3 recovered from the changes.

As it turns out, he was able to identify some places where ext3 failed to recover gracefully. Some corrupt directories failed to perform a consistency check, and so on. Eric posted a patch for these problems that made the recovery much more graceful. Eric and Steve discussed future possibilities for fsfuzzer, and Eric said he had also ran some fsfuzzer experiments on XFS, in which he had observed similar results. Pavel Machek mentioned that he'd played with a similar tool a few years ago, testing ext2. Apparently those experiments led to some e2fsck fixes as well. According to Pavel, ReiserFS and VFAT were too buggy at the time, to get any meaningful results from the tests. They just broke.

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.

New Non-Journaling Filesystem, SpadFS

Mikulas Patocka created SpadFS as part of his PhD thesis, and released it to the world, generating a big discussion primarily about Mikulas's design. SpadFS abandons filesystem journaling as too complex and prone to bugs, relying on a simpler method of avoiding data corruption called crash counting. Crash counting stores a variable that indicates whether the filesystem is mounted or unmounted; and associates that value with data as it is stored. In some cases, altered data is saved as a duplicate of existing data until a proper unmount occurs. This way, if the system ever crashes, SpadFS can detect the crash because its crash-count variable no longer matches the mount state; and so it can revert to the most recent consistent state of the filesystem.

Not everyone liked the SpadFS design, and some folks felt that Mikulas had misunderstood some fundamental concepts; but in the end it seems as though SpadFS has a fairly bright future. Even Linus offered a whole bunch of practical suggestions to help get it into the codebase quickly. He even paid it the high compliment of, "It doesn't look horrible to me." So, welcome to the kernel, SpadFS!

Promise Opens Some Hardware Specs

Jeff Garzik has been negotiating with Promise for permission to post programming specifications for some of their SATA hardware, and now they've agreed to release a part of this information for the first time. The PDC20319 is now an open chipset, as is 2037x and 205xx. This represents the entire line of chips supported by sata_promise.c.

This is really excellent news! Promise has been very reluctant until now to release any of their hardware specifications, so this represents a great leap forward into the open source world for them. Kudos to Promise. Hopefully, as Jeff puts it, these new docs will inspire some Linux hacker out there to make something efficient and useful for Linux.

Peace-Forking util-linux

Karel Zak remarked that the util-linux upstream project seemed to be neglected by its official maintainer, Adrian Bunk. As maintainer of the Red Hat util-linux package, Karel wanted to do something to get the upstream project moving again. His plans included creating a git repository and a web page for the project, inviting new developers to join in the effort, and starting merging patches from the various distributions.

His only concern was that he didn't like the idea of forking the code directly out from under Adrian, without at least some discussion. His hope was that Adrian would turn the reins over to him, and have a peaceful transition. However, H. Peter Anvin had some interesting things to say about whether or not to fork.

Peter's idea was that a fork could be a perfectly acceptable first step in some cases. He cited his own experience with tftp, where he initially forked the code into the tftp-hpa project; after awhile, the netkit maintainer invited him to be the official tftp maintainer, so tftp could be dropped from the core netkit code.

Peter argued that forking a project allowed a potential maintainer to prove their maintainership qualities before actually taking something over, which could put the current maintainer – even one who was too busy to do active work on the project, but still reluctant to part with it – more at ease and open to the idea of handing control over to someone who has already demonstrated that they can handle it.

It's an interesting idea. It's true that some forks are very violent and controversial, but probably most are not and take place under our noses all the time. Every single Linux distribution uses its own forked version of the kernel. And many kernel forks exist in parallel to Linus's tree. We don't think about these as forks so much, because no one is upset about them. It's only the really violent forks, like libc and emacs, that stick in our minds; but these may be the exception to a much more peaceful and amicable rule.

Reducing Pointless Warnings

Kernel compilation typically produces a massive number of warnings, but who cares as long as the process finishes? Torvalds wants to decrease the number of kernel warnings. Some warnings are actually useful and mean that some code is broken and should be fixed, but who can spot those in the mix?

Folks like Martin J. Bligh are already offering to submit patches to remove unneeded warnings. Other folks also started looking for ways to clear out warnings, but if it was an easy task, it wouldn't be a problem in the first place.

This strikes me as something that takes a while to resolve, because new code going into the kernel creates new warnings; but eventually extra warnings will be eliminated, and the code approval process will change. Eventually, I'd expect tight controls on patches that produce warnings; similar to the requirment that patches be accompanied by a "Signed-Off-By" line before being accepted. Folks may end up having to justify each warning in their changelog entry at submit time.

Ext4 In the Kernel

After much discussion and several attempts, the new ext4 code has been included in 2.6.19-rc2. The speed of inclusion can be attributed to the contributors responding to technical criticism, and producing code that keeps with the kernel style. This may have something to do with the fact that folks like Andrew Morton are involved.

Some people and projects have a more direct line into the kernel than others; and some folks would say this shows prejudice on the part of Torvalds and other top developers. I think that kernel development is so streamlined, and so many patches are accepted in a short time (hundreds went into -rc2 alone), that Torvalds and other main contributors give preference to people who are easier to work with. Trying to create a purely equal system would greatly decrease how many patches are accepted into new versions. On the flip side, people – even developers – can learn to be easy to work with.

Bypassing GPL Tainting

Alexey Dobriyan noticed that Linuxant's hsfmodem and hcfpcimodem used a well-known trick to get around the kernel's GPL test for kernel tainting. By including a null character in the license call, they've been tricking the kernel into believing it consists of fully GPLed code when in fact it may not. The call in question looks like this:

```
MODULE_LICENSE("GPL\0for files ⮐
in the \"GPL\" directory; for ⮐
others, only LICENSE file ⮐
applies");
```

This loophole has existed for quite awhile without being fixed. This time, Jan Engelhardt proposed a patch to ensure that no extra null characters are present in that call.

advertisement