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.

Finding bugs

Every once in awhile, someone discovers a really old bug, one that is very difficult to trigger, or that requires someone to intentionally try to take advantage of it. No less than two such bugs were uncovered this past March.

The first of these was not quite so ancient, but still fairly old, and dated back to the early days of Ingo Molnar's "O(1)" scheduler, now simply known as "the scheduler". Ingo's algorithm is such a vast improvement over the original, taking the same amount of time to handle any number of processes, that at this point there is simply no alternative.

Nonetheless, there had always been reports of some uneven interactive performance under his scheduler. Finally in March it was discovered that, due to a bug in the timing code, tasks that were CPU intensive and not particularly interactive, were being inadvertently marked as highly interactive, and given favored scheduling status. This in turn caused processes that really were interactive to appear jumpy and irregular.

Ingo found a fix for this, and now it is likely that the new scheduler will turn out to be even more of an improvement than people had originally suspected.

The second bug, discovered at around the same time by Andrzej Szombierski, was actually much more severe, and involved a security exploit that could allow any local user to gain root access to the machine. Not only that, but the bug was so old it affected all systems going back to kernel version 2.2. By a strange twist, 2.5 kernels appear not to be vulnerable. The characteristics of the bug were quite subtle, and the fixes ended up making changes to the way Linux behaved in various circumstances.

Alan Cox, who announced the security hole and the fixes, said he believed no actual software would be inconvenienced by the changes, and that the specific alterations would only affect unusual debugging situations. But of course, we all know that one of the features of open development is that once you say no one will be inconvenienced, you will immediately hear from multiple people who all require everything to remain exactly as it was. And that's just what happened.

A number of real-world scenarios came out of the woodwork to say they'd been broken by the proposed fixes. The most significant of these, perhaps, was UML (User-Mode Linux) in combination with some particular patches that had been floating around.

Matthew Grant had been using these to provide email services to a number of hospitals in Bangladesh. But aside from the question of how these various projects were going to deal with the proposed changes, was the question of whether the relevant Linux kernel maintainers (Alan and Marcelo Tosatti) should put out new kernel versions right away, or wait for their normal release date. Alan, maintaining 2.2, had perhaps the easier decision to make. He simply released 2.2.25 with only that fix, and pushed all other pending changes into the future 2.2.26 release.

Marcelo, on the other hand, did not release a fixed 2.4.21 kernel right away, and this became the subject of some controversy. Given the subtle nature of the exploit, the fact that it was "only" a local exploit and not subject to remote attacks, and the fact that a 2.4.21 kernel would come out with the fix in the normal course of events, folks were confused over what should be done.

But when the suggestion was made that people who really cared about the issue could get a security update from their favorite distribution vendor, some larger issues came up. Would vendor kernels eventually become the standard way for people to get sources? Would it be impractical to download the 'official' sources and compile by hand? The discussion went round and round, with some folks saying this particular bug was just not important enough to warrant a full release, and other folks saying that the official kernel sources should be the most cutting edge of all available source trees. ■

Spelling errors

Software developers are notoriously bad spellers. Most of the time this goes unnoticed, as long as they manage to spell the C keywords and variables correctly; but every now and then someone gets a bee in their bonnet about the sheer quantity of misspellings in the kernel source code comments and documentation.

This time, Dan Kegel wrote a script (mainly as a joke, if truth be known) to go through the source files and produce a report on spelling errors in the C comments.

Since software developers are also notorious for their ability to be obsessive over minutiae, this little script has received a tremendous amount of their attention.

Should British versions of words be preferred over American, or vice versa? What about the possibility of accidentally 'correcting' words that are actually variable names?

Linus Torvalds, among others, was thrilled to be able to clean up some of the more horrendous violations, and began accepting spelling patches as fast as Dan and others could send them in. This action has actually resulted in the introduction of a few new errors as well as the ongoing refinement of Dan's script. ■

Better BitKeeper

When Linus Torvalds decided to start using BitKeeper, a proprietary version control system, one aspect was that developers who chose not to use BitKeeper would have a harder time getting their patches accepted. Linus promised this wouldn't happen. But as of this past March, it has become clear that Linus has an easier time interfacing with BitKeeper users than with non-BitKeeper users. This is not entirely unexpected. BitKeeper is a very powerful tool that interoperates very well with itself.

A reasons Linus and others made such a big deal about not favoring BitKeeper users is because the implication would be that properly participating in kernel development would require a proprietary tool. Lately Linus has begun asking BitKeeper users to act as middlemen between him and non-BitKeeper users.

One of the arguments that is made by folks trying to replace BitKeeper, is that it would be much easier if they could just ignore or quickly work around certain problems, like the need for a truly distributed repository. Linus and BitKeeper's author, Larry McVoy, both claim that these problems are very significant and cannot be worked around.

The debate continues. On one side is a group of high-powered kernel developers who do not use BitKeeper, either because they reject proprietary software or because they are prevented by BitKeeper's own license; but who still want access to the BitKeeper repositories.

One great benefit of using BitKeeper has been that Linus's working tree has been made publicly available. Any developer may pull a current snapshot from Linus's own BitKeeper repository, and submit patches against that version. Non-BitKeeper users have had to wait a greater or lesser amount of time before gaining access to those same files.

The group of developers objected to this and started coding a clumsy alternative to BitKeeper, allowing them to read the repository and communicate via the secret BitKeeper networking protocol. With the threat of producing an actual BitKeeper replacement, they were able to force Larry to provide a real-time CVS tree that reflected the current state of Linus's repository.

Once this access had been granted, work on the free alternative (called BitBucket) apparently stopped. Of course, CVS is not at all up to the task of mirroring a BitKeeper repository, so Larry and his employees at BitMover went to the trouble to make all of BitKeeper's historical data and other metadata was preserved, either in the CVS history or as CVS comments.

These comments can be parsed and processed, to give developers access to individual 'changesets', allowing them to manage their own patches and ongoing development, without requiring BitKeeper on their local systems. ■

advertisement