Interview with Christoph Hellwig

RIK FARROW

Christoph Hellwig has been working with and on Linux for the last ten years, dealing with kernel-related issues much of the time. In addition he is or was involved with various other open source projects. After a number of smaller network administration and programming contracts, he worked for Caldera's German development subsidiary on various kernel and user-level aspects of the OpenLinux distribution. Since 2004 he has been running his own business, focusing on consulting, training, and contracting work in the open source hemisphere. Specializing on Linux file systems and storage, he is also active in bordering areas such as virtualization and networking. He has worked for well-known customers such as Dell, SGI, IBM, Red Hat and startups like LeftHand Networks and Smapper Applied Data or Nebula. hch@infradead.org

Rik Farrow is the editor of *;login:*. rik@usenix.org

I first noticed Christoph Hellwig during FAST conferences. With his spiky hair and non-academic background, he had a different presence from most attendees. But I could tell he was an expert when it came to XFS, a file system that was created by Silicon Graphics specifically for working with large files, then later ported to Linux.

Since we both attended Linux FAST, I got to know Christoph a bit better. Christoph is an independent consultant and a Linux kernel developer, a combination that is rare, as most Linux developers have some corporate sponsorship, usually in the form of a job. I also noticed that Christoph likes to ski, reminding me of the many ski bums I've met who arrange their life and work so they are free to ski when the slopes are best.

But unlike those ski bums, Christoph has established his chops as a Linux file system and kernel developer expert. I was curious about how he got to his unusual position, and started this interview after watching Christoph help run the Linux FAST '15 workshop (see summary in this issue).

Rik: When/how did you start working with Linux?

Christoph: I was in high school and learned about Linux from a magazine related to all things Internet that was popular in Germany in those days, and came with a ready-made Linux install CD. I installed it on my Dad's PC. Eventually, I got my own PC and had to fix the Linux sound driver for it, which was my first kernel contribution.

Rik: You worked for a while at Caldera, then started taking classes in Austria.

Christoph: I only worked for Caldera for about a year, maybe a year and a half, and only did so part time while finishing high school. Apparently I was productive enough that no one really knew I was a high school student. When Caldera shut down their German engineering office I got an offer from SGI to work full time for them out of their Munich office.

Math was interesting but a lot of work, so I switched to computer science. That turned out to be a bad idea, as I already had a background in computer science and now a fair knowledge of math, so the fairly low standards of the CS curriculum started to bore me. I was much happier working a real job where I learned from reading code, talking (or emailing) with experts in their field, and reading the bits of CS research that actually matter to me, which I wasn't exposed to at all at my university.

Through a loophole, I was taking lots of masters-level courses even though I never got an undergrad degree, but abusing those loopholes made me lose a lot of credits whenever they changed the curriculum, so in the end I had to give up my plan to slowly get my undergrad degree while working full time (and skiing full time!). In retrospect, trying to get that degree at all is probably the biggest regret I have, but that's definitively not the feeling I had back then.

Rik: When did you start working with file systems? Or how did you get involved? The first time I recall seeing you was at FAST, and you were making points about XFS. In my mind, I found myself labeling you the XFS guy.

Christoph: I started out my "file-system career" by hacking on support for simple foreign file systems early on in my Linux career—that's where my maintainer hat for the SysV / V7 file-system driver started as well. Then I moved on to play with the newly open-sourced IBM JFS file system and helped to clean it up so it could be merged into the Linux kernel. SGI hired me as junior file-system person with Linux community experience in 2002, and I gradually moved up the food chain from there.

Rik: During Linux FAST 2015, you and Ted Ts'o often mentioned tools that actually help make it easier to develop concurrent code inside of the kernel, rather than using FUSE. What are these tools, and how does one learn about using them?

Christoph: There are a few tools I rely on a lot:

- lockdep
  This is a tool to verify locking order, but also much more complicated constraints, such as locks taken in the page-out path but not held around memory allocation, which can lead to page outs. It also detects locks that are held while the containing objects are also holding locks, and much more. In a heavily multithreaded environment, this tool probably has the checks that code triggers most often.
- slab debugging
  The instrumented memory allocator that detects all kinds of use after frees is highly useful, although there are lots of libraries that provide the same functionality in user space these days. There also are the kmemleak and kasan tools that allow for very advanced memory leak detection.
- instrumented linked list / debug object
  We also have the instrumented version of linked lists, which is very helpful in avoiding corruption, and the debug object's functionality that allows you to enforce lifetime rules for arbitrary kernel objects.
- static trace points (trace events)
  Linux now has a very nice way of embedding trace points into code that has almost zero overhead if not enabled. Code that has a lot of those is really easy to instrument, especially since the tools for it allow nice use cases like only enabling the events for specific processes or various other conditions. The trace events also double as software profiling counters for the perf tool, allowing for very interesting semantic profiles.

But even more important than these are tools that help improve the code at runtime, most importantly the "sparse" tool, which allows us to, for example, annotate integer types as "big endian" or "little endian" and warn about missing conversions without forcing the use of complex types.

All the kernel debug options are documented sparsely in their help texts and the kernel documentation. For information more suitable to beginners, a Web search usually gives good results on sites such as stackoverflow.com.

Rik: Do you think it is harder to get involved as a Linux kernel developer now than it was when you started?

Christoph: I think doing the same work is easier these days with all the tools we have, but finding it might be harder, and becoming relevant definitively is.

Rik: You said you work a lot on the SCSI subsystem. Why is the SCSI subsystem so important to file systems?

Christoph: SCSI is one of the two standards most block storage devices speak today. SCSI is used over SAS, Fibre Channel, and iSCSI for enterprise storage; over USB for many consumer devices; and these days also in a lot of virtualized environments. Many operating systems, including Linux, also go through the SCSI layer to access the other major protocol, ATA, for historical reasons. In short, SCSI is what most block-based file systems end up using underneath, and it is one of the most important layers below the file system.

Rik: Your name was in the news in March 2015. You are involved in a legal action against VMware, because VMware has used Linux code in their hypervisor, as well as used Busybox as a shell [1, 2]. I'm wondering why you, someone who is self-employed and thus only making money when doing billable work, is taking the time to be involved with this lawsuit?

Christoph: Mostly, I want to avoid people piggybacking on my work, or the work of us kernel developers in general, without giving anything back.

Resources

[1] Software Freedom Conservancy, "Frequently Asked Questions about Christoph Hellwig's VMware Lawsuit": http://sfconservancy.org/linux-compliance/vmware-lawsuit-faq.html.

[2] Jonathan Corbet, "A GPL-Enforcement Suit against VMware: https://lwn.net/SubscriberLink/635290/e501ce0264c182f4/.