Recollections: An Interview with Doug Rabson (dfr@)

BY TOM JONES

The FreeBSD project started out with contributions from many hands, but the early days of the project and the people behind our favorite Operating System haven't been covered in much detail. As a part of FreeBSD's 30th Anniversary, I set out to speak to those involved at the start of development.

This installment is with Doug Rabson, who has been a FreeBSD committer since 1994 and is currently working on improving FreeBSD support for modern container orchestration systems such as podman and kubernetes.

TJ: Could you explain—generally—what you were up to in the late 1980s and early 1990s before the project?

DR: I graduated from college in the early 1980s. Before that, I had been exposed to BSD and we used it on some of the machines there, but I didn't pursue that kind of operating systems focus at all. I went to work for a little games company that some friends had started, and we told them that Unix is cool, that they should absolutely get Unix if they were going to buy a computer. They bought a MicroVAX, put Ultrix on it, which was 4.2 BSD—approximately—and we wrote Magnetic Scrolls. We were writing interactive fiction using Unix systems because the micros we were targeting were too weak.

I remember when the 4.3 BSD tapes were released, I got a copy from a friend at Imperial College, University of London.

That background interest stayed with me. I remember when the 4.3 BSD tapes were released, I got a copy from a friend at Imperial College, University of London. Just for curiosity, I wanted to understand how it worked. The idea of being able to read the source code was very cool. I read through it, figured out some of the pieces that were missing—and figured out that I didn't know enough to try to fill in the gaps.

A bit later, and this is again still just before FreeBSD, I heard about the 386BSD precursor to the Open/Net/FreeBSD group. Somebody did fill in the gaps—somebody who actually knew what they were doing. And I installed that on some scratch hardware I found at work. And it worked. It didn't work very well. I mean, it was sort of broken. And a bunch of people like Jordan—and I've forgotten all the names—Nathan Whitehorn, I think, David Greenman. Anyway, they got together with a set of patches for 386BSD, because the author wasn't really interested in doing much more with it. At that point, he'd written his article for Dr. Dobbs and had done the work. But he wasn't taking patches. There was a kind of organic movement to collate the patches to various bugs and features that had been added to 386BSD. And that was the 386BSD patch kit, which went through a number of versions. When it was clear that the author, Mr. Jolitz, wasn't really interested in taking his project further, that turned into the BSDs.

That was the point where Net and Free split. FreeBSD people wanted to target one viable platform, which is the 386, PC-based commodity hardware. The NetBSD people were interested in holding on to the portability that was always there in the BSD platform. And so, they diverged at that point. I don't think there were any hard feelings. It was just a difference in focus.

So that was what I had been doing up to the beginning of the project. I wasn't a contributor to any of the pieces, but rather an enthusiastic user of it.

TJ: How did you follow the discourse around the BSD developments?

DR: About that time, I was working for a little company we put together, and we arranged for a link to Usenet, which was a bulletin board system that predates a lot of classic internet stuff. I read quite a bit about things on there. You could follow along with the development in the Usenet groups that were related to BSD and similar things. I managed to sort out a method of getting email, which again, wasn't that easy at the time.

I found the mailing lists, and that was where I became properly informed about the project. Initially, I used that with a bit of wrangling to try to get things to work before ISPs existed in this country. I managed to get onto the mailing lists and then eventually an ISP did exist in this country, and I had dial-up internet, and it was off to the races after that.

TJ: How did you get the software before there were ISPs?

DR: So—dragging up old memories—I think some of it might have been posted to Usenet. I believe my employer did have some access. We certainly weren't on the Internet, but we had a connection to Usenet.

There was also a UK-based bulletin board called CompuServe where you could dial up and download stuff. That might have been part of what was going on, and that seems more plausible because Usenet was a bit of a Wild West situation, whereas with CompuServe, you could definitely download things. Yeah, I'm not absolutely certain how I first got hold of a copy of 386 BSD. I remember it being on about 10 or 15 floppies. I'm pretty sure that Usenet played a part. Once FreeBSD existed, there were some very useful mailing lists--some still exist today. There were, however, many fewer mailing lists than there are today. They were extremely useful in just keeping up and figuring out what people were doing, where the project was going, that kind of thing. That was in the FreeBSD 1 days.

TJ: What was the process of going from an enthusiastic user to a contributor?

DR: I was using FreeBSD. We had a little start-up that did 3D graphics technology. I put together a server for us to use for file sharing and email and things like that. By that time, we had dial-up. That was running FreeBSD 1.0, possibly 1.1.

In that company of about five or six people, we had one CD-ROM drive. I said, hey, we have a network. I can put that on the server, and we can share it via NFS. It was a bit of a can of worms that I opened there because it didn't work very well. One of the things that didn't work very well was that FreeBSD wasn't able to share the CD-ROM via NFS. I did some research and found some patches that somebody else had written. I had no idea who wrote them, to be honest. I applied the patches to our server. Yay, it worked.

I think the patches were originally for FreeBSD 1.0. I remember having to port them to FreeBSD 1.1 because there were some differences between the two releases. I think I sent my changed patch set to one of the main lists saying, hey, I ported this guy's work. It works on the current release. This is it. I think around that time, FreeBSD 2 was nearly happening.

The BSDI lawsuits were being resolved. One of the things we agreed to was to mothball the whole source tree from FreeBSD 1.0 and take a clean, legally-agreed copy of the 4.4 BSD Lite 2, I think, sources which everyone agreed definitely didn't include AT&T intellectual property. Then we forklifted the parts of FreeBSD that were clearly unencumbered and put together FreeBSD 2 from a clean base.

When I posted these patches to FreeBSD 1, the FreeBSD 2 thing was getting ready to happen. I think also at that point, my business partners were in California on a sales trip, and they happened to meet Jordan. I don't think that was particularly unusual because he knew the name of the company I was working for. It was in my email signature.

A friend of his was talking to my colleagues about 3D graphics. Jordan joined the meeting and the upshot was that he phoned me up and said, do you want to be a committer? At that point, I ported the things that I had been playing around with on FreeBSD 1 to FreeBSD 2 and got involved with that whole project of making FreeBSD 2 at least as good as 1.1x was—so moving a bunch of stuff from FreeBSD 1 to 2. That was when I started to be actively involved. It was FreeBSD 2.0 and beyond. That would have been 1994, I guess.

I ported the things that I had been playing around with on FreeBSD 1 to FreeBSD 2 and got involved with that whole project of making FreeBSD 2 at least as good as 1.1x was.

TJ: Then you went from being a committer to joining the first core team. How did that come about?

DR: From 1995 to 1997, I was working for Microsoft, and I didn't do much original work in FreeBSD at the time. Looking at my commit record, I was still somewhat actively doing things with NFS. I was fixing bugs and things, but not trying to do anything really interesting because I didn't want my employer to have rights to cool stuff for FreeBSD. Anyway, I didn't do much during that period of time. In 1997, I left Microsoft, and took some time away from paid work to properly connect with the project. I had some ideas going on in my head based on the way the Microsoft operating systems work. Things like loadable kernel modules, which were poorly supported in FreeBSD at the time, were hugely useful, still are.

I thought that model was much better than the giant kernel that contains everything, the model we were mostly using. I worked on the kernel linker. That took me to mid-1997, I guess. The idea came up that, hey, we've been working on this single platform, 386. We've done pretty well with it. We've got a stable operating system that people are able to use.

Should we consider a second platform? I don't know whose idea that was, but at one point, somebody from Digital offered us some loan hardware for DEC alpha. Jordan included me in that discussion and said, hey, do you want an alpha-based computer? Yeah, sure. DEC donated a bunch of hardware. We had this idea that we would port FreeBSD to this new platform. This is an interesting platform because, at least at the time, it looked like it could be viable as a commercial platform. The chips weren't crazily expensive.

The rest of the hardware in the machines was more or less PC-ish. We felt that this could be viable. It was a 64-bit platform, which was a necessary step for FreeBSD to take. We were already starting to approach the limitations of 32-bit platforms for some of our users. Alpha was there, and I got involved with that. Eventually, I ported the kernel with some help from NetBSD sources. That happened in 1998, mostly. As part of that work, I renovated the whole device driver architecture because alpha was different and needed an abstraction layer. I put that in and did a lot of work on it. In 1999, I went to the Usenix ATC to talk about my work with peers I'd never actually met. Everyone was just emailed at that point. Jordan grabbed me halfway through the conference and said, hey, do you want to join the core team? The core team had already existed for pretty much the whole time since FreeBSD's first release. I only joined it towards the end of the first core team. The first core team was before we did elections—that person looks like he's doing something interesting—let's grab it! That was how it was.

TJ: I don't know if elections are better.

DR: I think they are. I was kind of skeptical at the time, but there are lots of things I like about them. The term limits give you a clean point in time when you can say, hey, I don't want to deal with this level of involvement right now. I'm going to step back. I did that a bit later on. I got invited to the first core team, which was a fairly organic, self-organizing entity. From reading my old emails recently, I'm reminded that core team was very heavily technically focused. There was an architecture element to it that's intentionally not part of the remit for core these days. It was a bit different.

I think the first core grew from the patch kit folks. The people that were involved with the patch kit that ended up wanting to do the FreeBSD single platform focused on building something for people to use. Those people, by and large, were the first cohort in core zero. And then that group of people invited others. So, of course, that's how I entered the team, but I was toward the end of core zero. I don't recall exactly. I wasn't paying much attention to who was doing what.

TJ: How did the project change during your time on core zero and core one?

DR: I think the biggest thing that we changed was the election, and that was pushed by some members of core zero who wanted to clarify how the project was going to be governed—and have some bylaws. That was a huge change. I think it was a good change for bringing more of the project users into the project committers, at least, into the decision-making process. That was a cultural change, which I think was needed at the time. That was the end of core zero.

We got that process sorted out. I remember some meetings at Usenix and afterwards to nail down the details, get them agreed to by the membership, and then arranging the first election. I ran in the first election partly because I still wanted to be involved in the project at that level. I wanted the transition to an elected model to be successful, so a lot of us ran just so there were people who people already knew who were part of the election. The whole project, the whole thing would have failed if core zero said, yeah, here are the new rules--we're going off to the pub now. Yeah, so committing to the new model. I was part of that first election and was elected because I did have a reasonably high profile at the time. I was doing a lot of work on the kernel. I was making some significant changes. It felt natural for me to run because I was heavily involved.

I wanted the election system to succeed. It wasn't my idea, but I liked it once it was fleshed out. So, what changed during core one? That was 2000? Was that 2000 or 2001? I think that at that point, I started to get a bit burned out by the whole core thing. It was turning into a system for governing rather than a technical oversight. I found that more difficult to cope with than just figuring out what's broken and how to fix it.

We had some difficult decisions to make. This was during core zero, around Matt Dillon, and probably a few other things. I was just starting to get a little bit burned out, a bit crispy at least, on being part of the governance of the project rather than just being a contributor. That's more of a personal thing that changed. I'm struggling to think of anything tangible in the project that changed.

Yes, IA64 was interesting. We had a 64-bit platform, but it looked like Digital was going to drop the ball there.

TJ: We covered alpha a little bit, but what about the IA64 port?

DR: Yes, IA64 was interesting. We had a 64-bit platform, but it looked like Digital was going to drop the ball there. It was still being produced. That was probably after Compaq bought it out, and I thought the writing was on the wall for alpha, but people still needed a 64-bit platform. Yahoo, in particular, had some workloads that were running up against the limitations of the 32-bit address space, and they really wanted a 64-bit platform. I was at Usenix ATC in 2000. Paul Saab turned up and gave me a good six inches worth of technical documentation on IA64 and said, hey, you know how to port the kernel! Have a look at this. It wasn't clear that IA64 was the right direction to take, but it would take us closer to a somewhat x86-compatible, 64-bit platform. It was architecturally interesting. It did things in a different way, and I was curious about how that would work.

When I did alpha, a lot of it was helped by taking inspiration and code from NetBSD. They'd done the port a bit ahead of us. I wanted to do that process again but write it all myself just to prove to myself that I could do it. It wasn't just a question of getting some Lego pieces and putting them together and saying, hey, I did it. I wanted to build the pieces as well. I did that for IA64 and I had some great tooling to make it easier. I used simulations extensively in both ports to help get the thing up and running. Yahoo arranged for me to get some test hardware and I still have it. It's underneath my desk. It hasn't been switched on in 20 years. I got the test hardware and brought up system. I wasn't very impressed by the hardware.

Compared to the PC platforms I was using at the time, I thought this was going to be far too expensive. I couldn't see it running at scale. The goal in those ports in those days was can it build itself. Can it self-host? Can it build its own source code? I got it to that state. Along the way, I wanted to use some 386-only tools, those from Perforce that we were using for some private source code control. I didn't have an IA64 binary for that, so, I wrote the beginnings of what's now the previous 32-bit ABI. At that time, it was a 386 ABI hosted in my IA64 kernel. That code still gets used these days as the 32-bit compatibility layer.

I wrote enough of that to get Perforce to work, but I wasn't convinced that it would be a successful platform. It was a niche platform in the end, and it had its successes in that niche role. But I couldn't see anyone like Google or Yahoo or any of the other big internet players using it at scale, not with the hardware I'd seen. HP picked up Compaq and ended up being the main booster of IA64, because, I think, some of their IP went into Itanium from their PA risk architecture. HP was a big booster of the platform.

Another project member was working at HP, was interested in IA64, and I more or less let him take over development after about—I'm going to say 2001 or so—maybe 2002. I remember doing the 32-bit subsystems for IA64 in 2002. So, yeah, I kind of took it up to that point of being viable, self-hosting, then somebody else took it on.

I think they were both important ports for different reasons. The existence of IA64 made it easier for Peter Wemm to do the AMD64 port.

The legacy is having a truly free, self-supported, functioning operating system that doesn't involve license politics.

TJ: What is the lasting legacy of FreeBSD?

DR: The legacy is having a truly free, self-supported, functioning operating system that doesn't involve license politics, and when literally you can put in an embedded device, sell it, and nobody's going to start complaining on the internet that you haven't ticked the right boxes or released the source code of your thing. It's super easy for anyone to pick up the previous project. We've made it really clear where the boundaries are between simple copyright and complicated copyright parts of the system. So, I think that, yeah, that's a viable resource for embedded development for literally anything. You can find FreeBSD inside all kinds of weird stuff. It's the basis for a whole line of router hardware from people like Juniper. It's the control plane in a lot of storage appliances. FreeBSD used to be part of random internet firewall devices. It still is in all sorts of things that you just treat as an appliance.

They just work. And the reason they work is because FreeBSD is super easy to use, both legally and technically. And I think that's an important part of its legacy. There are almost certainly other things. I think that the quality of the code in FreeBSD makes it a positive resource for the rest of the operating system community. We do things in FreeBSD so that the ideas in FreeBSD can cross-pollinate other platforms and vice versa.

If FreeBSD was terrible, we wouldn't really be a part of that group of self-improving projects. This is the danger of monoculture. And FreeBSD is doing its part to avoid monoculture. And part of that is healthy cross-pollination. I get ideas from Linux. Hopefully Linux occasionally gets ideas from us. I know that they've taken some of our driver stuff in the past. So yeah, being a good partner in an ecosystem of similar projects is part of it.

TJ: Is there anything you would like to add?

DR: This last year, I have been re-connecting with the project after a fairly long period of low involvement. The main difference I see now is that we take a lot more care to avoid breaking things. Today, we have a decent unit test suite, continuous integration systems, and a growing culture of code review—compared to the early days when I would test my own changes on an ad-hoc basis, sometimes send them to people by email to look at, but not always. This is a good change overall since it reduces risk and tends to result in a stable platform, but it is a different (and slower) way of working and I think it can be hard to find the balance between minimizing risk and innovation.

TOM JONES is a FreeBSD committer interested in keeping the network stack fast.