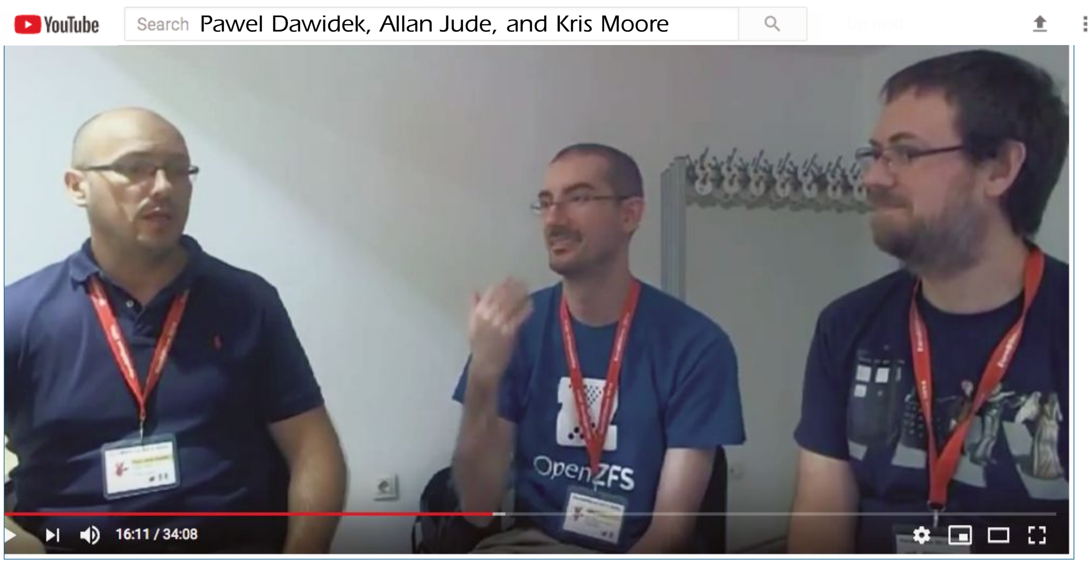
AN·INTERVIEW·WITH

Pawel Dawidek

by Allan Jude and Kris Moore

The *FreeBSD Journal* Editorial Board suggested that some of the outstanding interviews from the BSD Now series might be of interest to readers as they reflect the state of technology when the interview took place. Here, we've transcribed and excerpted from a 2014 Interview with Pawel Dawidek by Kris Moore and Allan Jude.



Kris Moore: BSD Now, Episode 62, *Gift from the Sun*. We're recording October 29, 2014. I'm your host, Kris Moore.

Allan Jude: And I'm Allan Jude.

Kris: We're going to be joined by Pawel Dawidek, who has done a lot of things in FreeBSD over the years, including the initial ZFS port—and, you know, that's kind of a big thing. So, we're going to hear about how that came out, what he's up to now, and a whole lot more. Pawel will talk about the work he did to port ZFS to FreeBSD. GEOM, GELI, Capsicum—various topics.

Allan: Yeah, it's not just that having ZFS ported to FreeBSD was a huge deal. But as he tells the story about how it actually happened, it really gives insight into how developing actually gets done.

Kris: That would have been a great interview itself but then we get GEOM, GELI, Capsicum, and all this other stuff too.

Allan: So, now we are joined by Pawel Dawidek from The FreeBSD Project. Thanks for joining us.

● **Pawel Dawidek**: *Certainly, thank you.*

Kris: The first question we ask pretty much everybody is what's your backstory, how did you first get into BSD?

● **Pawel**: *I started with computers when I was 12 years old, but at that time I was using C-64 and Amiga. And when I went away to study, I used Debian for maybe a few months at most. A friend introduced me to FreeBSD and, basically, I switched almost directly from Amiga to FreeBSD. One of my problems had been that I didn't really have any Windows experience.*

Kris: Is that a problem? [laughs]

● **Pawel**: *It is a bit of a problem when it comes to family and friends, because if you are in IT, they assume you must know this stuff. You must know how to install anti-virus programs and things like that.*

Kris: Except that also means you don't get phone calls in the middle of the night saying my Windows PC broke again!

● **Pawel**: *Yes. But I still got those questions. Of course, I loved FreeBSD but maybe not from Day 1 because it took me a few tries before I successfully installed FreeBSD for the first time. But since then it has been a great adventure.*

Allan: What are you working on currently with FreeBSD?

● **Pawel**: *I'm not that active anymore, but I still do some things mostly with Capsicum and other security projects—my main interest.*

Allan: The other question we traditionally ask is what are some of the things that are currently in FreeBSD that we can blame you for?

● **Pawel**: *Ah yes, probably quite a few of them over the years. I feel like I've been a FreeBSD committer for 10 years now.*

Kris: Wow!

● **Pawel**: *I mostly worked on storage and security, but my code is in many different places. In storage, I worked on the various GEOM improvements and GEOM classes, like gmirror, gstripe, gconcat, and a few of the other GEOM classes like GELI. One of the projects I had the most fun with was porting ZFS to FreeBSD. It was a very interesting experience.*

Kris: Since you've brought that up, what was your interest in porting ZFS? What brought that on initially?

● **Pawel**: *I remember the exact moment. During a vacation, a friend told me about ZFS. He was working on a few things, and they used Solaris a lot. He told me about the filesystem Sun [Microsystems] was working on, and about features like many datasets, end-to-end checksuming, compression, integrated volume manager, and it just sounded great. So, I was hoping that when the filesystem was created, somebody would port it to FreeBSD.*

But, of course, in FreeBSD, it was a curse with filesystems. We had UFS and we've had a lot of failed attempts in porting other filesystems to FreeBSD.

I know there was HFS, ReiserFS, read-only. We had XFS, which was also read-only.

Many of them were unfinished, and it was really starting to be a problem. So, UFS2 came along, which improved things a bit, but of course some people wanted to see ZFS.

Kris: Sure.

● **Pawel**: *So, basically, I wanted to see how hard it would be to port ZFS. The project was such fun that I got passionate about it. Basically in the first 10 days and 10 nights, I was working nonstop. I had so much adrenaline I could actually just keep working, and I was sleeping for maybe only 2 hours on some days.*

Kris: Wow.

● **Pawel**: *And after 10 days, I did have a read-write working prototype. It was—*

Kris: the fastest filesystem port ever. [laughs]

● **Pawel**: Yes, *but of course it wasn't all roses. It took many months—many more months actually—to get ZFS into committable shape for FreeBSD. But it was great fun. Definitely.*

Allan: What were some of the hurdles you had to overcome to get ZFS into FreeBSD and get it all working?

● **Pawel**: *ZFS was nicely implemented. It was very clean, and it was certainly helpful that most of the code could compile in userland. So, the code was prepared to be able to compile in the kernel and in userland—which entailed a lot of work to make the code portable.*

I had to create this very ugly layer—a compatibility layer with Solaris, which I'm not proud of, but it allowed us to keep the changes to a minimum. The difference between us and upstream was extremely helpful when porting new versions because, of course, I ported a very early version of ZFS and it was very dynamically developed at Sun. So, there were massive changes and this layer helped us move forward very quickly.

ZFS is a filesystem, so it attaches to the VFS layer, which I find one of the most complex pieces of FreeBSD's kernel code. Dealing with VFS is tough.

Kris: I spent a week some months ago looking at Fuse and the VFS layer there.

● **Pawel**: *I'm a fan of simplifying VFS because our VFS is trying to help filesystem developers by doing a lot of the work for them—centralization and buffering and stuff like that. Which, in theory, is nice, but when you deal with a more complex filesystem, you do want to have more stuff moved to the filesystem itself.*

Allan: Right, you want to have more control as some strategy for buffering is different.

● **Pawel**: Exactly. *ZFS can be much more efficient when it comes to performance. It has more control over the whole process.*

Allan: Were there any other interesting things that came up when making Solaris code work on FreeBSD?

● **Pawel**: *Well, there were some differences, but surprisingly, the kernels' APIs were pretty similar. I think there is much less difference between FreeBSD and the Solaris kernel than between FreeBSD and Linux kernel. So that was helpful.*

Kris: Makes sense. This question is from our producer, and he's asking what would it finally take to get the native ZFS encryption?

● **Pawel**: *I'm not a fan of how ZFS encryption is implemented in the Oracle version.*

Allan: I don't think anybody is.

● **Pawel**: *Well, some people are. The idea behind Oracle encryption is that you encrypt individual datasets. But where do you want to use encryption? Mostly on your laptops because you take laptops with you, less so on servers because where do you store keys to allow for automatic boot?*

And how many users are on a laptop? Mostly one. And Oracle ZFS encryption assumes that there are multiple users, and they want to optimize this. So, every single dataset can be encrypted with user keys, which is problematic on many levels. First of all, it is totally incompatible with the deduplication because if you have different datasets using different keys, you have different datasets in the pool, so we cannot deduplicate.

The only way to deduplicate is to clone filesystems so the key is also cloned, which is not very nice.

Allan: Yeah, it is for separate users.

● **Pawel**: *Yes, exactly. So, I'm not a fan of this idea. I don't think it's really the way to go. The way I would implement ZFS encryption is to actually use ZFS encryption below the dataset level—encrypt raw data at the vdev level. This way we can also encrypt more data because, of course, with Oracle ZFS encryption we cannot encrypt zpool-wide metadata.*

Allan: Right. So please talk about the GELI stuff and what you've done with that system.

● **Pawel**: *GELI is basically a disk encryption; it resides below filesystems. There are many similar projects. Of course, Linux has its own implementation. GELI is obviously a long-running project.*

We did have disk encryption in FreeBSD when I started to work on GELI, but it didn't really fit my needs. So, I started to work on my own disk encryption, which does provide some unique features. For example, I don't know of any disk encryption software that provides data authentication.

And GELI was also, from Day 1, integrated with our opencrypto framework, so it can use crypto accelerators. GELI also provides many useful features—for example, you can use multiple secrets to decrypt your disk, partition, or any storage device. You can have a passphrase and a key stored on a pendrive.

You can also have multiple keys. You can imagine a scenario where in a company, your security officer has a different key to your master key, and you have your own key. So, if your employee loses his key to the data, the data is not lost, as there is an alternative key.

Allan: That's very helpful.

Kris: Definitely. Are you still actively involved in GEOM or any filesystem development?

● **Pawel**: *Currently I've mostly moved to security stuff. I do like to observe ZFS development these days. I am very happy that there are more and more companies and people involved because for a few years I think I was the only person who was porting ZFS to FreeBSD. And it was definitely not a one-person project.*

Allan: Yes.

● **Pawel**: *So, I waited for a long time for others to join, and there are quite a lot of companies actually using ZFS in production or for their products. So, when people are involved in ZFS development on FreeBSD, we have very good relations with the illumos community. The TRIM support was one of the developments we did in* FreeBSD. *I did it initially for Multiplay from the U.K. And this is something that only FreeBSD has as far as I know. There are many stories that companies are switching from Linux to FreeBSD to get ZFS. So that's very nice.*

I'm sure many of them were not sure at first but after they switched, I heard many positive stories.

Allan: Would you tell us about your work on Capsicum and what you're doing with that now.

● **Pawel**: *Well, the kernel part of Capsicum is more or less very usable. The thing we still work on is Casper daemon. It's a daemon that helps when using Capsicum because Capsicum provides a very tight sandbox. You have no access to global namespaces. This is actually great technology.*

It's really a great idea and security framework. In the past, I was working on my own security stuff—where it was just a bad idea to monitor system calls and stuff like that. Capsicum is the way to go, and it gets much more traction. I hope that Capsicum will be—

Kris: the way of the future.

● **Pawel**: Yes, and more and more useful.

Kris: Sure.

● **Pawel**: *Casper daemon is a way to help develop applications. For example, once you are in Capsicum sandbox, you are not able to resolve host names and Casper is there to enable you to do that. We are experimenting with Casper as a separate daemon, but we are learning it may not be the best idea, as this separate process is running in a different context, with its own resource limits, cpuset, routing table, etc. We may want to spawn Casper from the application process and inherit its context.*

Allan: Yes, so that it doesn't actually provide the easier way around resource limits for the CPU site or whatever.

● **Pawel**: *Exactly. There is some work to do on that front as well.*

Kris: Thanks. What exactly is your day job and do you use FreeBSD there?

● **Pawel**: *Oh, yes, we do a lot. I'm running my own company, Wheel Systems [now Fudo Security]. We are a security vendor; we provide security products. It turns out that the technologies I've been working on for FreeBSD over the years are very useful in our latest product—Fudo, where we use GELI to encrypt* *all the storage; we use ZFS to replicate the data between cluster nodes; we use compression and snapshots. And we heavily use Capsicum to make it all secure.*

We want to be sure that even if someone breaks into a single session, he cannot access other sessions. He cannot actually access anything, because if he breaks in before authentication, he won't be granted access to connect to the server. Only after successful authentication will we provide a connection to the destination server.

And Capsicum makes it really clean and very efficient actually.

Allan: You don't have to enumerate all the things you can't do. You're saying you're only allowed to do these things?

● **Pawel**: *Yes. This is capability ideology. You only grant the exact rights or access to resources that the process requires. Which is not UNIX ideology because, of course, if you are running a UNIX program, it has access to everything.*

Allan: Was there anything else you wanted to talk about?

● **Pawel**: *Not really.* ●

FreeBSD®

The FreeBSD Project is looking for

- Programmers
- Testers
- Researchers
- Tech writers
- Anyone who wants to get involved

Find out more by

Checking out our website
freebsd.org/projects/newbies.html

Downloading the Software
freebsd.org/where.html

We're a welcoming community looking for people like you to help continue developing this robust operating system. Join us!

Already involved?

Don't forget to check out the latest grant opportunities at
freebsdfoundation.org

Help Create the Future. Join the FreeBSD Project!

FreeBSD is internationally recognized as an innovative leader in providing a high-performance, secure, and stable operating system.

Not only is FreeBSD easy to install, but it runs a huge number of applications, offers powerful solutions, and cutting edge features. The best part? It's FREE of charge and comes with full source code.

Did you know that working with a mature, open source project is an excellent way to gain new skills, network with other professionals, and differentiate yourself in a competitive job market? Don't miss this opportunity to work with a diverse and committed community bringing about a better world powered by FreeBSD.

The FreeBSD Community is proudly supported by

FreeBSD FOUNDATION