

# Re: Porting OpenBSD's sysctl hw.sensors framework to FreeBSD

---

*Source:* <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2007-07/msg00064.html>

---

- *From:* Alfred Perlstein <[alfred@xxxxxxxxxxx](mailto:alfred@xxxxxxxxxxx)>
  - *Date:* Wed, 11 Jul 2007 11:14:20 -0700
- 

\* Robert Watson <[rwatson@xxxxxxxxxxx](mailto:rwatson@xxxxxxxxxxx)> [070711 11:05] wrote:

On Wed, 11 Jul 2007, Alfred Perlstein wrote:

Possibly, one way to do that is to provide a well thought out userland library that can make for a nice interface. If by doing it userland the kernel implementation can be kept smaller and more simple it might be a win.

That said, it may be a loss if you wind up having to duplicate work over and over for different devices it may be a loss.

Remember, once the kernel interface is exposed it has to be there almost forever, while a userland interface and much more easily be adapted.

The flip side of the user/kernel, of course, is that userland implementations may require more privilege to invoke than kernel implementations, since they may need to frob with /dev/pci, /dev/mem, etc, rather than accessing a very narrow sysctl interface using essentially unprivileged access. Compare, for example, the old ps(1) which required root or kmem privilege in order to be able to grope through kernel memory, and the new ps(1) that uses a set of controlled APIs that not only let it run as any user, but also scope the process information exposed by requesting process.

However, the point I think that you specifically are making is that if we define a user API to access the information, then the implementation can be flexible, and what we really want to know is how applications will interact with the data. I think this is precisely the right point — what we should not do is lock ourselves into a particular sysctl representation of the data and approach, which has happened to quite a few of our monitoring interfaces and makes it quite hard to abstract things, change the implementation, etc. Consider directly accessing sysctls (as done in most of netstat) and the slightly more abstract libkvm interfaces, which allow you to access information as data and hide the method (allowing the method

Re: Porting OpenBSD's sysctl hw.sensors framework to FreeBSD  
to be /dev/kmem, sysctls, coredumps, etc).

Yes, this was the point I was trying to put across, well said.

—  
– Alfred Perlstein

---

freebsd-arch@xxxxxxxxxxx mailing list  
<http://lists.freebsd.org/mailman/listinfo/freebsd-arch>  
To unsubscribe, send any mail to "freebsd-arch-unsubscribe@xxxxxxxxxxx"