Subject: Re: [Ksummit-2010-discuss] checkpoint-restart: naked patch
Posted by Jose R. Santos on Thu, 18 Nov 2010 20:13:27 GMT
View Forum Message <> Reply to Message

On Thu, 18 Nov 2010 10:48:34 +0100
Tejun Heo <tj@kernel.org> wrote:

> Hello, Pavel.
>
> On 11/18/2010 10:13 AM, Pavel Emelyanov wrote:
> >>> By this do you mean the very idea of having CR support in the
> >>> kernel? Or our design of it in the kernel?
> >>
> >> The former, I'm afraid.
> >
> > Can you elaborate on this please?
>
> I think I already did that several times in this thread but here's an
> attempt at summary.

Yet the arguments seem to be vague enough not to be convincing to the
people working on the code.

> * It adds a bunch of pseudo ABI when most of the same information is
> available via already established ABI.

Can you elaborate on this? What established ABI are you proposing we
use here. Hopefully we can turn this into a more technical discussion.

> * In a way which can only ever be used and tested by CR. If possible,

So what if it can only be tested with CR as long as we can make CR work
on a variety of environments? Scalability changes for _really_ large
SMP boxes can only be reliably tested by people such equipment. We are
not imposing any such restriction and this code can be tested on very
wide range of setups.

> kernel should provide generic mechanisms which can be used to
> implement features in userland. One of the reasons why we'd like to
> export small basic building blocks instead of full end-to-end
> solutions from the kernel is that we don't know how things will
> change in the future. In-kernel CR puts too much in the kernel in a
> way too inflexible manner.
>
> * It essentially adds a separate complete set of entry/exit points for
> a lot of things, which makes things more error prone and increases
> maintenance overhead across the board.

I partially agree with you here. There will be maintenance overhead every time you add code to the kernel that _may_ make changes in the future more complicated. This true for _any_ code that is added to the core kernel. Now in my experience such maintenance burden is most disruptive when the code being added creates a lot of new state that need to be tracked in multiple places unrelated to CR (in this case). Our argument is that the CR code is not creating new state that will cause painful future changes to the kernel. If you have specific example that you are concerned with, great. Lets discuss those.

Are we promising zero maintenance cost? But guess what, neither do most features that make into the kernel.

Now, if we change the argument around... What would be the maintenance cost keeping this outside the kernel. I would argue that it is much higher and would use SystemTap as the first example that come to mind.

> * And, most of all, there are userland implementation and
> virtualization, making the benefit to overhead ratio completely off.

Can we keep virtualization out of this. Every time someone mentions virtualization as a solution, it makes me feel like these people just don't understand the problem we are trying to solve. It is just not practical to create a new VM for every application you want to CR. These are two different tools to attack two different problems.

> Userland implementation _already_ achieves most of what's necessary
> for the most important use case of HPC without any special help from

What are these _most_ important cases of HPC that you are referring too? Can we do a lot of these cases from userspace? Sure, but why are the ones that can't be done from userspace any less important. If nobody cared about those, we would not be having this conversation.

> the kernel. The only reasonable thing to do is taking a good look
> at it and finding ways to improve it.

The userspace vs in-kernel discussion has been done before as multiple people have already said in this thread. Show me a version of userspace CR that can correctly do all that an in-kernel implementation is capable of.

> Thanks.
>

--
Jose R. Santos

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containe rs