Subject: Re: [RFC][PATCH 0/2] pipe: checkpoint and restart for pipe; brief description and test interface
Posted by serue on Wed, 31 Jan 2007 15:11:36 GMT
View Forum Message <> Reply to Message

Quoting Eric W. Biederman (ebiederm@xmission.com):
> Masahiko Takahashi <masahiko@linux-foundation.org> writes:
>
> > Hi everyone,

Thanks, Masahiko, for getting the discussion on how we want to
c/r going with a patchset!

> > I would like to post a small patch that implements pipe checkpoint
> > and restart functionality.
> >
> > The patch supports pipe buffers used with splice/vmsplice systemcall.
> > Usually, spliced buffers' pages are in memory cache because their data
> > are read from an filesystem. However the patch doesn't treat buffers'
> > pages as cached memory when restoring. If trying to treat as cached
> > memory, each buffer's page should be tagged with its filename when
> > checkpointing. This is not a difficult implementation, but I'm still
> > wondering this is worth implementing...
>
> Rewind a little bit please. I can clearly see where checkpoint/restart
> fit into the containers conversation.
>
> However I don't see where this checkpoint/restart patchset fits into
> the checkpoint restart picture.

I agree without a basic infrastructure to initiate checkpoints it
feels a little out of place, but presumably we can test this with any
of the existing c/r solutions out there?

> What ideas are you trying to discuss?
>
> I will say that I am not yet convinced that we need the kernel saving
> and restoring the checkpoint.

Some things will make sense to just do in userspace, but other things,
i.e. certainly remapped memory, and, perhaps pipes, should have help from
the kernel.

Or can the program driving the checkpoint just cat the pipe fd to get
the unread contents out to save in the checkpoint? If not, an
alternative might be to just add an f_op 'checkpoint', which for pipes
spits out unread contents, and for ordinary files just spits out pos
and flags... Though one problem with that is how to tell which two

processes were sharing the pipe? pipefile_fops->checkpoint() spits out
the pipefs inode number?

> Regardless of the approach when we start seriously discussing checkpoint/restart
> one of the big issues is how do we maintain an ABI to user space for all of
> the data so we can migrate applications across kernel version.

I had planned on waiting until pidspace was more complete to start that
discussion, but by all means lets start talking about a preferred
mechanism and api now.

Dave, does this pipe c/r module conflict with how you wer seeing memory
checkpoint happen?

Thanks again, Masahiko.

-serge

_____
Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers