Subject: Re: [Ksummit-2010-discuss] checkpoint-restart: naked patch
Posted by Gene Cooperman on Sun, 07 Nov 2010 23:31:31 GMT
View Forum Message <> Reply to Message

On Sun, Nov 07, 2010 at 04:44:20PM -0500, Oren Laadan wrote:
> [cc'ing linux containers mailing list]
>
> On 11/06/2010 04:40 PM, Gene Cooperman wrote:
>
> >8.  What happens if the DMTCP coordinator ( checkpoint control process) dies?
> > >   [ The same thing that happens if a user process dies.  We kill the whole
> > >     computation, and restart.  At restart, we use a new coordinator.
> > >     Coordinators are stateless. ]
>
> My experience is different:
>
> I downloaded dmtcp and followed the quick-start guide:
> (1) "dmtcp_coordinator" on one terminal
> (2) "dmtcp_checkpoint bash" on another terminal
>
> Then I:
> (3) pkill -9 dmtcp_coordinator
> ... oops - 'bash' died.
>
> I didn't even try to take a checkpoint :(

You're right.  I just reproduced your example.  But please remember that
we're working in a design space where if any process of a computation
dies, then we kill the computation and restart.  It doesn't matter to us
if it's a user process or the DMTCP coordinator that died.  I do think
this is getting too detailed for the LKML list, but since you bring it
up, here is the analysis.  The user bash process exits with:

[31331] ERROR at dmtcpmessagetypes.cpp:62 in assertValid; REASON='JASSERT(strcmp (
DMTCP_MAGIC_STRING,_magicBits ) == 0) failed'
    _magicBits =
Message: read invalid message, _magicBits mismatch.  Did DMTCP coordinator die uncleanly?

This means that when the DMTCP coordinator died, it sent a message to the
checkpoint thread within the user process.  The message was ill-formed.
The current DMTCP code says that if a checkpoint thread receives an
ill-formed message from the coordinator, then it should die.  It's not
hard to change the protocol between DMTCP coordinator and checkpoint
thread of the user process into a more robust protocol with RETRY, further
ACK, etc.  We haven't done this.  Right now, the user simply restarts from
the last checkpoint.  If one process of a computation has been compromised
(either DMTCP coordinator or user process), then the whole computation
has been compromised.  I think in a previous version of DMTCP, the policy

was to allow the computation to continue when the coordinator dies.
Policies change.

But I think you're missing the larger point. We've developed DMTCP over six years, largely with programmers who are much less experienced than the kernel developers. Yet DMTCP works reliably for many users. I consider this a credit to the DMTCP design. The Linux C/R design is also excellent.

Can we get back to questions of design, using the implementations as reference implementations? If you don't object, I'll also skip replying to the other post, since I think we're getting too detailed. I'm having trouble keeping up with the posts. :-) An offline discussion will give us time to look more carefully at these issues, and draw more careful conclusions.

Thanks,
- Gene

_____

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