Subject: Re: Checkpoint/restart (was Re: [PATCH 0/4] - v2 - Object creation with a specified id)
Posted by Oren Laadan on Thu, 17 Jul 2008 23:16:10 GMT
View Forum Message <> Reply to Message

Serge E. Hallyn wrote:
> Quoting Dave Hansen (dave@linux.vnet.ibm.com):
>> On Wed, 2008-07-09 at 18:58 -0700, Eric W. Biederman wrote:
>>> In the worst case today we can restore a checkpoint by replaying all of
>>> the user space actions that took us to get there. That is a tedious
>>> and slow approach.
>> Yes, tedious and slow, *and* minimally invasive in the kernel. Once we
>> have a tedious and slow process, we'll have some really good points when
>> we try to push the next set of patches to make it less slow and tedious.
>> We'll be able to describe an _actual_ set of problems to our fellow
>> kernel hackers.
>>
>> So, the checkpoint-as-a-corefile idea sounds good to me, but it
>> definitely leaves a lot of questions about exactly how we'll need to do
>> the restore.
>
> Talking with Dave over irc, I kind of liked the idea of creating a new
> fs/binfmt_cr.c that executes a checkpoint-as-a-coredump file.
>
> One thing I do not like about the checkpoint-as-coredump is that it begs
> us to dump all memory out into the file. Our plan/hope was to save
> ourselves from writing out most memory by:
>
> 1. associating a separate swapfile with each container
> 2. doing a swapfile snapshot at each checkpoint
> 3. dumping the pte entries (/proc/self/)
>
> If we do checkpoint-as-a-coredump, then we need userspace to coordinate
> a kernel-generated coredump with a user-generated (?) swapfile snapshot.
> But I guess we figure that out later.

I'm not sure how this approach integrates with (a) live migration (and
the iterative process of sending over memory modified since previous
iteration), and (b) incremental checkpoint (where except for the first
snapshot, additional snapshots only save what changed since the previous
one).

Oren.

>
> -serge

_____
Containers mailing list