Subject: Re: [RFD][PATCH] memcg: Move Usage at Task Move
Posted by KAMEZAWA Hiroyuki on Tue, 10 Jun 2008 08:11:26 GMT
View Forum Message <> Reply to Message

On Tue, 10 Jun 2008 14:50:32 +0900 (JST)
yamamoto@valinux.co.jp (YAMAMOTO Takashi) wrote:

> > 3. Use Lazy Manner
> > When the task moves, we can mark the pages used by it as
> > "Wrong Charge, Should be dropped", and add them some penalty in the LRU.
> > Pros.
> > - no complicated ones.
> > - the pages will be gradually moved at memory pressure.
> > Cons.
> > - A task's usage can exceed the limit for a while.
> > - can't handle mlocked() memory in proper way.
> >
> > 4. Allow Half-moved state and abandon rollback.
> > Pros.
> > - no complicated ones in the code.
> > Cons.
> > - the users will be in chaos.
>
> how about:
>
> 5. try to move charges as your patch does.
> if the target cgroup's usage is going to exceed the limit,
> try to shrink it. if it failed, just leave it exceeded.
> (ie. no rollback)
> for the memory subsystem, which can use its OOM killer,
> the failure should be rare.
>

Hmm, allowing exceed and cause OOM kill ?

One difficult point is that the users cannot know they can move task
without any risk. How to handle the risk can be a point.
I don't like that approarch in general because I don't like "exceed"
status. But implementation will be easy.

> > After writing this patch, for me, "3" is attractive. now.
> > (or using Lazy manner and allow moving of usage instead of freeing it.)
> >
> > One reasone is that I think a typical usage of memory controller is
> > fork()->move->exec(). (by libcg ?) and exec() will flush the all usage.
>
> i guess that moving long-running applications can be desirable
> esp. for not so well-designed systems.

>

hmm, for not so well-designed systems....true.
But "5" has the same kind of risks for not so well-desgined systems ;)

Thanks,
-Kame

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers