
Subject: Re: Fork bomb limitation in memcg WAS: Re: [PATCH 00/11] kmem controller for memcg: stripped down ve

Posted by [Frederic Weisbecker](#) on Wed, 27 Jun 2012 12:29:27 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Wed, Jun 27, 2012 at 01:29:04PM +0400, Glauber Costa wrote:

> On 06/27/2012 01:55 AM, Andrew Morton wrote:

> >>I can't speak for everybody here, but AFAIK, tracking the stack through
> >>the memory it used, therefore using my proposed kmem controller, was an
> >>idea that good quite a bit of traction with the memcg/memory people.
> >>So here you have something that people already asked a lot for, in a
> >>shape and interface that seem to be acceptable.

> >

> >mm, maybe. Kernel developers tend to look at code from the point of
> >view "does it work as designed", "is it clean", "is it efficient", "do
> >I understand it", etc. We often forget to step back and really
> >consider whether or not it should be merged at all.

> >

> >I mean, unless the code is an explicit simplification, we should have
> >a very strong bias towards "don't merge".

>

> Well, simplifications are welcome - this series itself was
> simplified beyond what I thought initially possible through the
> valuable comments
> of other people.

>

> But of course, this adds more complexity to the kernel as a whole.
> And this is true to every single new feature we may add, now or in
> the
> future.

>

> What I can tell you about this particular one, is that the justification
> for it doesn't come out of nowhere, but from a rather real use case that
> we support and maintain in OpenVZ and our line of products for years.

Right and we really need a solution to protect against forkbombs in LXC.
The task counter was more simple but only useful for our usecase and
defining the number of tasks as a resource was considered unnatural.

So limiting kernel stack allocations works for us. This patchset implements
this so I'm happy with it. If this is more broadly useful by limiting
resources others are interested in, that's even better. I doubt we are
interested in a solution that only concerns kernel stack allocation.
