Subject: Re: [PATCH v2 09/11] memcg: propagate kmem limiting information to children
Posted by Greg Thelen on Wed, 22 Aug 2012 01:09:03 GMT
View Forum Message <> Reply to Message

On Tue, Aug 21 2012, Michal Hocko wrote:

> On Tue 21-08-12 13:22:09, Glauber Costa wrote:
>> On 08/21/2012 11:54 AM, Michal Hocko wrote:
> [...]
>> > But maybe you have a good use case for that?
>> >
>> Honestly, I don't. For my particular use case, this would be always on,
>> and end of story. I was operating under the belief that being able to
>> say "Oh, I regret", and then turning it off would be beneficial, even at
>> the expense of the - self contained - complication.
>>
>> For the general sanity of the interface, it is also a bit simpler to say
>> "if kmem is unlimited, x happens", which is a verifiable statement, than
>> to have a statement that is dependent on past history.
>
> OK, fair point. We shouldn't rely on the history. Maybe
> memory.kmem.limit_in_bytes could return some special value like -1 in
> such a case?
>
>> But all of those need of course, as you pointed out, to be traded off
>> by the code complexity.
>>
>> I am fine with either, I just need a clear sign from you guys so I don't
>> keep deimplementing and reimplementing this forever.
>
> I would be for make it simple now and go with additional features later
> when there is a demand for them. Maybe we will have runtimg switch for
> user memory accounting as well one day.
>
> But let's see what others think?

In my use case memcg will either be disable or (enabled and kmem limiting enabled).

I'm not sure I follow the discussion about history. Are we saying that once a kmem limit is set then kmem will be accounted/charged to memcg. Is this discussion about the static branches/etc that are autotuned the first time is enabled? The first time its set there parts of the system will be adjusted in such a way that may impose a performance overhead (static branches, etc). Thereafter the performance cannot be regained without a reboot. This makes sense to me. Are we saying that kmem.limit_in_bytes will have three states?

- kmem never enabled on machine therefore kmem has never been enabled
- kmem has been enabled in past but is not effective is this cgroup (limit=infinity)
- kmem is effective in this mem (limit=not-infinity)