Subject: Re: [PATCH v3 04/13] kmem accounting basic infrastructure
Posted by Tejun Heo on Sun, 30 Sep 2012 08:47:50 GMT
View Forum Message <> Reply to Message

Hello, Michal.

On Thu, Sep 27, 2012 at 05:09:50PM +0200, Michal Hocko wrote:
> On Thu 27-09-12 07:33:00, Tejun Heo wrote:
> > I'm not too convinced. First of all, the overhead added by kmemcg
> > isn't big.
>
> You are probably talking about memory overhead which is indeed not that
> big (except for possible side effects as fragmentation which you mention
> bellow). But the runtime overhead is present, as far as I understand from
> what Glauber said. But, on the other hand, it is fair to say that those
> who _want_ use the feature should pay for it.

Yeah, as long as the overhead is reasonable and it doesn't affect non-users, I think we should put more emphasis on simplicity. cgroup is pretty hairy (from both implementation and interface POVs) to begin with. Unfortunately, what's reasonable or how much more emphasis varies widely depending on who one asks.

> > The hot path overhead is quite minimal - it doesn't do much more than
> > indirecting one more time. In terms of memory usage, it sure could
> > lead to a bit more fragmentation but even if it gets to several megs
> > per cgroup, I don't think that's something excessive. So, there is
> > overhead but I don't believe it to be prohibitive.
>
> Remember that users do not want to pay even "something minimal" when the
> feature is not needed.

Yeah but, if we can get it down to, say, around 1% under most workloads for memcg users, it is quite questionable to introduce full hierarchical configuration to allow avoiding that, isn't it?

> > The distinction between "trusted" and "untrusted" is something
> > artificially created due to the assumed deficiency of kmemcg
> > implementation.
>
> Not really. It doesn't have to do anything with the overhead (be it
> memory or runtime). It really boils down to "do I need/want it at all".
> Why would I want to think about how much kernel memory is in use in the
> first place? Or do you think that user memory accounting should be
> deprecated?

But you can apply the same "do I need/want it at all" question to the configuration parameter too. I can see your point but the decision

seems muddy to me, and if muddy, I prefer to err on the side of being
too conservative.

> > This is userland visible API.
>
> I am not sure which API visible part you have in mind but
> kmem.limit_in_bytes will be there whether we go with global knob or "no
> limit no accounting" approach.

I mean full hierarchical configuration of it. It becomes something
which each memcg user cares about instead of something which the base
system / admin flips on system boot.

> > We can always expand the flexibility. Let's do the simple thing
> > first. As an added bonus, it would enable using static_keys for
> > accounting branches too.
>
> While I do agree with you in general and being careful is at place in
> this area as time shown several times, this seems to be too restrictive
> in this particular case.
> We won't save almost no code with the global knob so I am not sure
> what we are actually saving here. Global knob will just give us all or
> nothing semantic without making the whole thing simpler. You will stick
> with static branches and checkes whether the group accountable anyway,
> right?

The thing is about the same argument can be made about .use_hierarchy
too. It doesn't necessarily make the code much harier. Especially
because the code is structured with that feature on mind, removing
.use_hierarchy might not remove whole lot of code; however, the wider
range of behavior which got exposed through that poses a much larger
problem when we try to make modifications on related behaviors. We
get a lot more locked down by seemingly not too much code and our long
term maintainability / sustainability suffers as a result.

I'm not trying to say this is as bad as .use_hierarchy but want to
point out that memcg and cgroup in general have had pretty strong
tendency to choose overly flexible and complex designs and interfaces
and it's probably about time we become more careful especially about
stuff which is visible to userland.

Thanks.

--
tejun