Subject: Re: [PATCH v3 04/13] kmem accounting basic infrastructure
Posted by Tejun Heo on Wed, 26 Sep 2012 23:33:34 GMT
View Forum Message <> Reply to Message

Hello,

On Thu, Sep 27, 2012 at 03:20:27AM +0400, Glauber Costa wrote:
> > One of the things wrong with that is that it exposes the limitation of
> > the current implementation as interface to userland, which is never a
> > good idea. In addition, how is userland supposed to know which
> > workload is shared kmem heavy or not? Details like that are not even
> > inherent to workloads. It's highly dependent on kernel implementation
> > which may change any day. If we hit workloads like that the right
> > thing to do is improving kmemcg so that such problems don't occur, not
> > exposing another switch.
>
> Sorry, there is nothing implementation dependent in here. One of the
> biggest consumers of all this, are dentries. Dentries are related to
> the paths you touch. If you touch files in a self-contained directory,
> where you don't expect anyone else to touch, this can safely be
> considered local. If you touch files all around, this can safely be
> considered not local. Where is the implementation dependent part?

For things like dentries and inodes and if that really matters, we
should be able to account for the usage better, no? And frankly I'm
not even sold on that usecase. Unless there's a way to detect and
inform about these, userland isn't gonna know that they're doing
something which consumes a lot of shared memory even that activity is
filesystem walking. You're still asking userland to tune something
depending on parameters not easily visible from userland. It's a lose
lose situation.

> >> Even after the limit is set, it is set only by workloads that want kmem
> >> to be tracked. If you want to track it during the whole lifetime of the
> >> cgroup, you switch it before you put tasks to it. What is so crazy about it?
> >
> > The fact that the numbers don't really mean what they apparently
> > should mean.
> >
>
> This is vague. The usage file in the cgroup means how much kernel memory
> was used by that cgroup. If it really bothers you that this may not be
> set through the whole group's lifetime, it is also easily solvable.

Yes, easily by having a global switch which can be manipulated when
there's no children. It really seems like a no brainer to me.

> > So, this seems properly crazy to me at the similar level of

> > use_hierarchy fiasco. I'm gonna NACK on this.
>
> As I said: all use cases I particularly care about are covered by a
> global switch.
>
> I am laying down my views because I really believe they make more sense.
> But at some point, of course, I'll shut up if I believe I am a lone voice.
>
> I believe it should still be good to hear from mhocko and kame, but from
> your point of view, would all the rest, plus the introduction of a
> global switch make it acceptable to you?

The only thing I'm whining about is per-node switch + silently
ignoring past accounting, so if those two are solved, I think I'm
pretty happy with the rest.

Thanks.

--
tejun
