Subject: Re: [PATCH v5 2/2] decrement static keys on real destroy time
Posted by akpm on Wed, 16 May 2012 21:13:42 GMT
View Forum Message <> Reply to Message

On Fri, 11 May 2012 17:11:17 -0300
Glauber Costa <glommer@parallels.com> wrote:

> We call the destroy function when a cgroup starts to be removed,
> such as by a rmdir event.
>
> However, because of our reference counters, some objects are still
> inflight. Right now, we are decrementing the static_keys at destroy()
> time, meaning that if we get rid of the last static_key reference,
> some objects will still have charges, but the code to properly
> uncharge them won't be run.
>
> This becomes a problem specially if it is ever enabled again, because
> now new charges will be added to the staled charges making keeping
> it pretty much impossible.
>
> We just need to be careful with the static branch activation:
> since there is no particular preferred order of their activation,
> we need to make sure that we only start using it after all
> call sites are active. This is achieved by having a per-memcg
> flag that is only updated after static_key_slow_inc() returns.
> At this time, we are sure all sites are active.
>
> This is made per-memcg, not global, for a reason:
> it also has the effect of making socket accounting more
> consistent. The first memcg to be limited will trigger static_key()
> activation, therefore, accounting. But all the others will then be
> accounted no matter what. After this patch, only limited memcgs
> will have its sockets accounted.

So I'm scratching my head over what the actual bug is, and how
important it is. AFAICT it will cause charging stats to exhibit some
inaccuracy when memcg's are being torn down?

I don't know how serious this in in the real world and so can't decide
which kernel version(s) we should fix.

When fixing bugs, please always fully describe the bug's end-user
impact, so that I and others can make these sorts of decisions.