Subject: Re: Re: [PATCH v5 00/10] per-cgroup tcp memory pressure
Posted by Glauber Costa on Fri, 18 Nov 2011 19:39:03 GMT
View Forum Message <> Reply to Message

On 11/17/2011 07:35 PM, David Miller wrote:
> From: James Bottomley<jbottomley@parallels.com>
> Date: Tue, 15 Nov 2011 18:27:12 +0000
>
>> Ping on this, please. We're blocked on this patch set until we can get
>> an ack that the approach is acceptable to network people.
>
> __sk_mem_schedule is now more expensive, because instead of short-circuiting
> the majority of the function's logic when "allocated<= prot->sysctl_mem[0]"
> and immediately returning 1, the whole rest of the function is run.

Not the whole rest of the function. Rather, just the other two tests.
But that's the behavior we need since if your parent is on pressure, you
should be as well. How do you feel if we'd also provide two versions for
this:
1) non-cgroup, try to return 1 as fast as we can
2) cgroup, also check your parents.

That could be enclosed in the same static branch we're using right now.

> The static branch protecting all of the cgroup code seems to be
> enabled if any memory based cgroup'ing is enabled. What if people use
> the memory cgroup facility but not for sockets? I am to understand
> that, of the very few people who are going to use this stuff in any
> capacity, this would be a common usage.

How about we make the jump_label only used for sockets (which is basic
what we have now, just need a clear name to indicate that), and then
enable it not when the first non-root cgroup is created, but when the
first one sets the limit to something different than unlimited?

Of course to that point, we'd be accounting only to the root structures,
but I guess this is not a big deal.

> TCP specific stuff in mm/memcontrol.c, at best that's not nice at all.

How crucial is that? Thing is that as far as I am concerned, all the
memcg people really want the inner layout of struct mem_cgroup to be
private to memcontrol.c This means that at some point, we need to have
at least a wrapper in memcontrol.c that is able to calculate the offset
of the tcp structure, and since most functions are actually quite
simple, that would just make us do more function calls.

Well, an alternative to that would be to use a void pointer in the newly

added struct cg_proto to an already parsed memcg-related field
(in this case tcp_memcontrol), that would be passed to the functions
instead of the whole memcg structure. Do you think this would be
preferable ?

> Otherwise looks mostly good.

Thank you for your time.