
Subject: Re: [RFC 0/7] Initial proposal for faster res_counter updates
Posted by [Glauber Costa](#) on Fri, 30 Mar 2012 10:46:17 GMT
[View Forum Message](#) <> [Reply to Message](#)

> Note: Assume a big system which has many cpus, and user wants to divide
> the system into containers. Current memcg's percpu caching is done
> only when a task in memcg is on the cpu, running. So, it's not so dangerous
> as it looks.

Agree. I actually think it is pretty

> But yes, if we can drop memcg's code, it's good. Then, we can remove some
> amount of codes.

>

>> But the cons:

>>

>> * percpu counters have signed quantities, so this would limit us 4G.

>> We can add a shift and then count pages instead of bytes, but we

>> are still in the 16T area here. Maybe we really need more than that.

>>

>

>

> struct percpu_counter {

> raw_spinlock_t lock;

> s64 count;

>

> s64 limites us 4G ?

>

Yes, I actually explicitly mentioned that. We can go to 16T if we track
pages

instead of bytes (I considered having the res_counter initialization code to
specify a shift, so we could be generic).

But I believe that if we go this route, we'll need to either:

1) Have our own internal implementation of what percpu counters does

2) create u64 accessors that would cast that to u64 in the operations.

Since it

is a 64 bit field anyway it should be doable. But being doable
doesn't mean we

should do it....

3) Have a different percpu_counter structure, something like struct
percpu_positive_counter.

>

>> * some of the additions here may slow down the percpu_counters for

>> users that don't care about our usage. Things about min/max tracking

>> enter in this category.

>>

>

>
> I think it's not very good to increase size of percpu counter. It's already
> very big...Hm. How about
>
> struct percpu_counter_lazy {
> struct percpu_counter pcp;
> extra information
> s64 margin;
> }
> ?

Can work, but we need something that also solves the signedness problem.
Maybe we can use a union for that, and then stuff things in the end of a
different
structure just for the users that want it.
