Subject: Re: [PATCH v2 3/5] change number_of_cpusets to an atomic
Posted by Christoph Lameter on Tue, 24 Apr 2012 18:27:09 GMT
View Forum Message <> Reply to Message

On Tue, 24 Apr 2012, Glauber Costa wrote:

> > > It doesn't seem to be the case here.
> >
> > How did you figure that? number_of_cpusets was introduced exactly because
> > the functions are used in places where we do not pay the cost of calling
> > __cpuset_node_allowed_soft/hardwall. Have a look at these. They may take
> > locks etc etc in critical allocation paths
> I am not arguing that.
>
> You want to avoid the cost of processing a function, that's fair.
> (Note that by "function call cost" I don't mean the cost of processing a
> function, but the cost of a (potentially empty) function call.)
> The real question is: Are you okay with the cost of a branch + a global
> variable (which is almost read only) fetch?

No and that is why the static branching comes in. It takes away the global read of the number_of_cpusets variable in the critical paths.

> The test of a global variable can - and do as of right now - avoid all the
> expensive operations like locking, sleeping, etc, and if you don't need to
> squeeze every nanosecond you can, they are often simpler - and therefore
> better - than static branching.

Better than static branching? This is in critical VM functions and reducing the cache footprint there is good for everyone.

> Just to mention one point I am coming across these days - that initiated all
> this: static patching holds the cpu_hotplug.lock. So it can't be called if you
> hold any lock that has been already held under the cpu_hotplug.lock. This will
> probably mean any lock the cpuset cgroup needs to take, because it is called -
> and to do a lot of things - from the cpu hotplug handler, that holds the
> cpu_hotplug.lock.

Transitions from one to two cpusets are rare and are only done when a cpuset is created in the /dev/cpuset hierachy). You could move the code modification outside of locks or defer action into an event thread if there are locks in the way.