Subject: Re: [PATCH] Mark timer_stats as incompatible with multiple pid namespaces
Posted by ebiederm on Thu, 13 Dec 2007 21:48:31 GMT

Ingo Molnar <mingo@elte.hu> writes:

> * Eric W. Biederman <ebiederm@xmission.com> wrote:
>
>> >> Well struct pid * works in that case if you grab the reference to
>> >> it.
>> >
>> > but the display of the stats might happen much later. The point of
>> > this API is to save pid+comm, which gives users a good idea about
>> > what caused the events in the past - without having to pin any
>> > resource of that task.
>>
>> Likewise struct pid is designed not to be a problem if pinned. It is a
>> little heavier then it used to be with the addition of pid namespace
>> support but not much. And if it is to heavy struct pid needs to be
>> fixed.
>>
>> Holding the struct pid very much does not pin the task struct, and it
>> shouldn't pin any other resources. I agree 64bytes or so is a bit
>> more to pin then 4 bytes but it really isn't a lot.
>
> yeah, and i have no conceptual objections - i just wanted to outline the
> thinking behind /proc/timer_stats.

Sure. Appreciated. Outlining the thinking in the other direction
struct pid is supposed to be the pid representation in the kernel.
All of the pid_t stuff really should be pushed as close to the
kernel/user boundary as possible. The closer I get to that ideal
the more cases I find that we need to handle like /proc/timer_stats
and the closer we get to having a complete pid namespace. <pant pant pant>

A struct pid is also now all you pin (besides the inevitable file,
dentry, inode trio) when you open a directory in /proc. So the
task can be freed. Which removed some low-mem exhaustion scenarios
when we introduced it.

Eric

____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers