Subject: Re: [PATCH] kthread: Spontaneous exit support
Posted by ebiederm on Mon, 23 Apr 2007 17:45:51 GMT
View Forum Message <> Reply to Message

Oleg Nesterov <oleg@tv-sign.ru> writes:

> On 04/23, Christoph Hellwig wrote:
>>
>> On Sun, Apr 22, 2007 at 09:12:55PM -0600, Eric W. Biederman wrote:
>> >
>> > This patch implements the kthread helper functions kthread_start
>> > and kthread_end which make it simple to support a kernel thread
>> > that may decided to exit on it's own before we request it to.
>> > It is still assumed that eventually we will get around to requesting
>> > that the kernel thread stop.
>>
>> I don't think having to parallel APIs is a good idea, people will
>> get utterly confused which one to use. Better always grab a reference
>> in kthread_create and drop it in kthread_stop. For normal thread
>> no change in behaviour and only slightly more code in the slowpath.
>>
>> Of course it will need an audit for half-assed kthread conversion
>> first to avoid task_struct reference count leaks.
>
> In that case it is better to grab a reference in kthread(). This also
> close the race when a new thread is woken (freezer) and exits before
> kthread_create() does get_task_struct().
>
>> In addition to that kthrad_end implementation look wrong. When
>> the kthread has exited prematurely no one will call complete
>> on kthread_stop_info.done before it's been setup.
>
> This is not true anymore, see another patch from Eric
>
> kthread-enhance-kthread_stop-to-abort-interruptible-sleeps.patch

Ok. Thinking about it I agree with Christoph that parallel API's can
be a problem.

However we do still need to support kernel threads where kthread_stop will
never be called. There appear to be a few legitimate cases where
someone wants to fire off a thread and have it do some work but don't
care at all for stopping it before it is done.

So I propose we add a kthread_orphan as a basic primitive to decrement the
count on the task_struct if we want a kthread to simply exit after it
has done some work.

And as a helper function we can have a kthread_run_orphan.

I think having a kthread_orphan will document what we are doing better
and make it easy to find kernel threads that don't use kthread_stop.

The pain is that this requires an audit of all kernel kthread creators
so that we call kthread_orphan on the right ones, or else we will have
a task_struct leak. At least that isn't a fatal condition.

Eric

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