

---

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](mailto: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>

---