Subject: Re: [RFC][PATCH] rename 'struct pid'
Posted by Dave Hansen on Wed, 11 Apr 2007 02:17:34 GMT
View Forum Message <> Reply to Message

On Tue, 2007-04-10 at 19:28 -0600, Eric W. Biederman wrote:
> Dave Hansen <hansendc@us.ibm.com> writes:
>
> > I've been hacking quite a bit on the pidspace code. I've run
> > into a bug or two, and had a heck of a time debugging it.
> > Other than my inferior puny monkey brain, I'm directing some
> > of the blame squarely in the direction of the 'struct pid'.
> >
> > We have pid_t, pid_ns, struct pid and pid_link, at _least_.
> > Seeing code like get_pid(pid->pid_ns->pid_type[PIDTYPE_PID])
> > is mind-numbing to say the least.
>
> get_pid(pid->pid_ns->pid_type[PIDTYPE_PID]) is complete and utter
> nonsense.

Guilty as charged. It was utter nonsense.

> > It makes it really hard to comprehend, and even harder to debug.
>
> Given that you quoted nonsense I can understand the comprehension
> problem.

Here are five actual examples that I see in my tree right now. Each has
the string "pid" occur in it at least 4 times:

```
  struct pid_nr *pid_nr = alloc_pid_nr(ppid_nr->pid_ns, new_pid);

        hlist_add_head_rcu(&pid->pid_chain, &pid_hash[pid_hashfn(pid->nr)]);

        pid_nr = alloc_pid_nr(ppid_nr->pid_ns, new_pid);

        struct pid_nr *pid_nr = alloc_pid_nr(ppid_nr->pid_ns, new_pid);

        hlist_add_head_rcu(&pid->pid_chain, &pid_hash[pid_hashfn(pid->nr)]);
```

What I'm telling you is that I really think that this naming is making
it hard to enhance and debug this code.

> > We honestly have a lot of code like this:
> >
> > pid = pid_nr(filp->f_owner.pid);
> >
> > WTF? It's getting a pid from a pid? Huh? :)
>

> Clearer would be:
>
> user_pid = pid_to_user(filp->f_owner.pid);

Yes. Having a user vs. kernel process id distinction would be nice.
The fact remains, though, that we call the numbers pids in userspace.
We deal with processes very differently in the kernel, and we can and
should make that distinction in the naming of our structures.

> > It makes sense when you go look at the structures, but
> > sitting in the middle of a function and when you can't see
> > all of the struct declarations can be really sketchy.
> >
> > So, I propose that we rename the one structure that seems to
> > be the focus of the problem: 'struct pid'.
>
> NAK.

The pidspace stuff is taking a lot longer than it should to develop.
I'm not quite sure why, but I have the feeling that this readability
impediment is part of it.

> > Fundamentally, it
> > is a 'process identifier': it helps the kernel to identifty
> > processes. However, as I noted, 'pid' is a wee bit overloaded.
> >
> > In addition to "identifying" processes, this structure acts
> > as an indirection or handle to them. So, I propse we call
> > these things 'struct task_ref'.
>
> Renaming the structure above doesn't help and the structure represents
> a pid in a more fundamental way then pid_t can.

I completely agree. Thus, I think we should get away from the 'pid'
nomenclature and move to something a bit more descriptive. The 'struct
pid' is much more useful than a pid_t, and we should try to
differentiate it in some way. Calling it a 'pid' really doesn't do it
justice.

> A pid (pid_t or
> struct pid) isn't just an identfier it is a handle to processes.
> struct pid just does so more directly because it is inside the kernel.

Let's face it, "pid" has a meaning. It's a number. It's what you
kill(1). The meaning has been there for a long, long time. 'struct
pid' is a completely different concept, and it's certainly more than
"just a number".

So, please consider that there are actual kernel developers hacking on
this stuff who are having problem with it. The function of 'struct pid'
is great, it's a wonderful concept. It just needs a slightly different
name in order to more easily relate that concept to those that are
trying to use it.

If anyone can think of some more incremental ways to do this, or has
other suggestions on how to make it more clear, I'm all ears.

-- Dave