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

Dave Hansen <hansendc@us.ibm.com> writes:

> On Wed, 2007-04-11 at 10:13 -0600, Eric W. Biederman wrote:
>> Because the suggestions I have seen are based on a partial understanding
>> of struct pid, and the suggested renames will reinforce misunderstandings
>> of what struct pid is.
>
> Let's take a step back, then and see if we can nicely define what it
> does. Maybe we can come up with a bit better comment for 'struct
> pid'.
>
> ---
>
> 'struct pid' is a way for the kernel to get (and keep) a reference to a
> task (or set of tasks) with a particular numeric value. If there are
> any tasks with this numeric value from now on, the structure will stay
> pointed to those tasks.
>
> In general, the underlying task will not change, but it can in cases
> like when a non-thread-leader does an exec() and takes over as the
> leader (the pid will change).
>
> This reference will not keep the tasks from exiting. But, even if all
> of these tasks exit, the kernel will honor the fact that your reference
> exists and will not re-create any tasks with the same numeric value, as
> long as you allow your reference to persist.

The user space pid values do and should get recycled even if you are holding
a reference to a struct pid. Otherwise there are some easy denial of service
attacks by opening /proc/<child> directories and then having your child exit.

...

A struct pid is very similar to a list symbol. Two of them can be compared for
equality just by comparing pointers. You can get different information about
a struct pid by examining different ''slots''. A struct pid has a
name, but unlike lisp the name is numeric instead of a string.

Furthermore there is a one to one mapping from all valid user space
pid numbers (in a single namespace) to a kernel struct pid. Whether
you compare the pointer is kernel space or the numeric user space
value you will get the same answer.

The kernel struct pid can however live on after it's numeric name is

no longer visible to user space.

The kernel struct pid is an identifier even if it isn't numeric.

The point of struct pid is so you can operate on pids without having
to worry about the pesky user space numeric names.

Eric

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