Subject: Remaining straight forward kthread API conversions...
Posted by ebiederm on Thu, 19 Apr 2007 06:52:28 GMT
View Forum Message <> Reply to Message

The following patches are against 2.6.21.rc6-mm1.
Hopefully that is enough to catch most of the recent development
activity.

I am aiming to remove all kernel threads that handle signals
from user space, to remove all calls to daemonize and kernel_thread
from non-core kernel code.

kernel thrreads handling signals from user space is a problem because
it makes the kernel thread part of the user/kernel API which make
changing things difficult and it breaks as soon as you are inside of
a pid namespace because you won't be able to see your kernel thread.

Calling kernel_thread has problems because it returns a pid_t value
which once we get to the pid namespace is context depending so it
cannot be used to globally identify a process. kernel_thread is
also a problem because it traps user space state and requires us
to call daemonize to free that state.

daemonize is a maintenance problem because every time you play with
user space state and limiting things you need to remember to update
daemonize. Occasionally it has taken years like in the case of the
mount namespace before someone realizes they need to update it.
With the kthread api we no longer need daemonize.

In addition we don't want kernel threads visible in anything but
the initial pid namespace or they will hold a reference to a
child pid namespace. However calling kernel_thread from a non-kernel
parent in a child pid namespace will give the thread a pid in
the child pid namespace, and there is nothing daemonize can do about
it. So daemonize appears impossible to support going forward, and
I choose to remove all of it's callers rather than attempt to support
it.

Eric