Subject: Re: [RFC][PATCH] fork: Don't special case CLONE_NEWPID for process or sessions
Posted by Pavel Emelianov on Thu, 01 Nov 2007 09:28:23 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:

Sorry for the late answer, I have just noticed that I forgot to
answer on this patch.

> Given that the kernel supports sys_setsid we don't need a special case
> in fork if we want to set: session == pgrp == pid.
>
> The historical (although not 2.6) linux behavior has been to start the
> init with session == pgrp == 0 which is effectively what removing this
> special case will do.

Hm... I overlooked this fact. Looks like the namespace's init will
have them set to 1.

> Is there any reason why we want/need this special case in fork? Or

Mainly to address the issue I describe below.

> can we remove it and save some code, make copy_process easier to read
> easier to maintain, and possibly a little faster?
>
> I know it is a little weird belong to a process groups that isn't
> visible in your pid namespace, but it there are no good reasons
> why it shouldn't work.

This is not good to have such a situation as the init will have
the ability to kill the tasks from the namespace he can't see,
e.g. his parent and the processes in that group.

> I think making this change makes the interface more flexible,
> and general.
>
> Signed-off-by: Eric W. Biederman <ebiederm@xmission.com>
> ---
> kernel/fork.c | 18 +++++----------
> 1 files changed, 5 insertions(+), 13 deletions(-)
>
> diff --git a/kernel/fork.c b/kernel/fork.c
> index ddafdfa..b0de799 100644
> --- a/kernel/fork.c
> +++ b/kernel/fork.c
> @@ -1292,20 +1292,12 @@ static struct task_struct *copy_process(unsigned long clone_flags,

```
>    if (thread_group_leader(p)) {
>     if (clone_flags & CLONE_NEWPID) {
>      p->nsproxy->pid_ns->child_reaper = p;
> -    p->signal->tty = NULL;
> -    set_task_pgrp(p, p->pid);
> -    set_task_session(p, p->pid);
> -    attach_pid(p, PIDTYPE_PGID, pid);
> -    attach_pid(p, PIDTYPE_SID, pid);
> -   } else {
> -    p->signal->tty = current->signal->tty;
> -    set_task_pgrp(p, task_pgrp_nr(current));
> -    set_task_session(p, task_session_nr(current));
> -    attach_pid(p, PIDTYPE_PGID,
> -      task_pgrp(current));
> -    attach_pid(p, PIDTYPE_SID,
> -      task_session(current));
>     }
> +   p->signal->tty = current->signal->tty;
> +   set_task_pgrp(p, task_pgrp_nr(current));
> +   set_task_session(p, task_session_nr(current));
> +   attach_pid(p, PIDTYPE_PGID, task_pgrp(current));
> +   attach_pid(p, PIDTYPE_SID, task_session(current));
>
>     list_add_tail_rcu(&p->tasks, &init_task.tasks);
>     __get_cpu_var(process_counts)++;
```