Subject: Re: [RFC][PATCH 07/16] Move alloc_pid call to copy_process
Posted by xemul on Thu, 24 May 2007 08:35:48 GMT
View Forum Message <> Reply to Message

> Index: lx26-21-mm2/kernel/pid.c
> ==========
> --- lx26-21-mm2.orig/kernel/pid.c 2007-05-22 16:59:46.000000000 -0700
> +++ lx26-21-mm2/kernel/pid.c 2007-05-22 17:06:48.000000000 -0700
> @@ -216,6 +216,10 @@ fastcall void free_pid(struct pid *pid)
> /* We can be called with write_lock_irq(&tasklist_lock) held */
> unsigned long flags;
>
> + /* check this here to keep copy_process() cleaner */
> + if (unlikely(pid == &init_struct_pid))
> + return;
> +

This looks ugly to me.
That's the same as if we put
if (ns == &init_pid_ns)
return;
in put_pid_ns() call.

Such small struts of their own do not introduce any noticeable
effect, but when we have them in many places (and on fast patsh
like alloc_pid()) the performance hurts...

> spin_lock_irqsave(&pidmap_lock, flags);
> hlist_del_rcu(&pid->pid_chain);
> spin_unlock_irqrestore(&pidmap_lock, flags);
> @@ -224,12 +228,16 @@ fastcall void free_pid(struct pid *pid)
> call_rcu(&pid->rcu, delayed_put_pid);
> }
>
> -struct pid *alloc_pid(void)
> +struct pid *alloc_pid(enum copy_process_type copy_src)
> {
> struct pid *pid;
> enum pid_type type;
> int nr = -1;
>
> + /* check this here to keep copy_process() cleaner */
> + if (unlikely(copy_src == COPY_IDLE_PROCESS))
> + return &init_struct_pid;
> +
> pid = kmem_cache_alloc(pid_cachep, GFP_KERNEL);
> if (!pid)

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

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