Subject: Re: [RFC][PATCH 07/16] Move alloc_pid call to copy_process
Posted by Sukadev Bhattiprolu on Thu, 24 May 2007 09:42:41 GMT
View Forum Message <> Reply to Message

Pavel Emelianov [xemul@sw.ru] wrote:
| sukadev@us.ibm.com wrote:
| > Pavel Emelianov [xemul@sw.ru] wrote:
| > | > 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.
| >
| > I agree about the ugly part :-) but we need to distinguish
| > between idle thread and normal thread at some point in do_fork().
|
| Why not keep this as it was - pass the pid from do_fork() or
| do_fork_idle(). Why is that bad?

When CLONE_NEWPID flag is specified and we create a new pid ns
we need the task_struct that will become the child reaper for
the new ns. The task_struct is allocated in copy_process().

We could make a second check for CLONE_NEWPID in copy_process()
and then initialize the reaper for the ns, but that could have
same or more performance impact ?

|
| > | 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...
| >
| > I agree and we have been trying to keep the impact as low as possible.
|

| In this patches - yes. But when we have many patches with sucn
| "hooks" this becomes noticeable and hard to debug.
|
| > |
| > |
| > | > 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
| > | >
| > | > _____
| > | > Devel mailing list
| > | > Devel@openvz.org
| > | > https://openvz.org/mailman/listinfo/devel
| > | >
| >

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