Subject: [PATCH 5/6] Use task_pid() to find leader's pid
Posted by Sukadev Bhattiprolu on Fri, 06 Jul 2007 05:55:52 GMT

Subject: [PATCH 5/6] Use task_pid() to find leader's pid

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>

Use task_pid() to get leader's pid since find_pid() cannot be used
after detach_pid(). See comments in the code below for more details.

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
---
 fs/exec.c | 9 ++++++++-
 1 file changed, 8 insertions(+), 1 deletion(-)

Index: lx26-22-rc6-mm1/fs/exec.c
===
--- lx26-22-rc6-mm1.orig/fs/exec.c 2007-07-05 18:56:53.000000000 -0700
+++ lx26-22-rc6-mm1/fs/exec.c 2007-07-05 18:58:32.000000000 -0700
@@ -905,10 +905,17 @@ static int de_thread(struct task_struct
    * The old leader becomes a thread of the this thread group.
    * Note: The old leader also uses this pid until release_task
    *   is called. Odd but simple and correct.
+   * Note: With multiple pid namespaces, active pid namespace of
+   *   a process is stored in its struct pid. The detach_pid
+   *   below frees the struct pid, so we will have no notion
+   *   of an active pid namespace until we complete the
+   *   subsequent attach_pid(). Which means - calls like
+   *   find_pid()/pid_to_nr() return NULL and cannot be used
+   *   between the detach_pid() and attach_pid() calls.
   */
  detach_pid(tsk, PIDTYPE_PID);
  tsk->pid = leader->pid;
- attach_pid(tsk, PIDTYPE_PID, find_pid(tsk->pid));
+ attach_pid(tsk, PIDTYPE_PID, task_pid(leader));
  transfer_pid(leader, tsk, PIDTYPE_PGID);
  transfer_pid(leader, tsk, PIDTYPE_SID);
  list_replace_rcu(&leader->tasks, &tsk->tasks);

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

Subject: Re: [PATCH 5/6] Use task_pid() to find leader's pid
Posted by Pavel Emelianov on Fri, 13 Jul 2007 05:22:26 GMT

sukadev@us.ibm.com wrote:
> Subject: [PATCH 5/6] Use task_pid() to find leader's pid
>
> From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
>
> Use task_pid() to get leader's pid since find_pid() cannot be used
> after detach_pid(). See comments in the code below for more details.
>
> Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>

Acked-by: Pavel Emelianov <xemul@openvz.org>

> ---
> fs/exec.c | 9 ++++++++-
> 1 file changed, 8 insertions(+), 1 deletion(-)
>
> Index: lx26-22-rc6-mm1/fs/exec.c
> ===
> --- lx26-22-rc6-mm1.orig/fs/exec.c 2007-07-05 18:56:53.000000000 -0700
> +++ lx26-22-rc6-mm1/fs/exec.c 2007-07-05 18:58:32.000000000 -0700
> @@ -905,10 +905,17 @@ static int de_thread(struct task_struct
>   * The old leader becomes a thread of the this thread group.
>   * Note: The old leader also uses this pid until release_task
>   *   is called. Odd but simple and correct.
> +   * Note: With multiple pid namespaces, active pid namespace of
> +   * a process is stored in its struct pid. The detach_pid
> +   * below frees the struct pid, so we will have no notion
> +   * of an active pid namespace until we complete the
> +   * subsequent attach_pid(). Which means - calls like
> +   * find_pid()/pid_to_nr() return NULL and cannot be used
> +   * between the detach_pid() and attach_pid() calls.
> +   */
>   detach_pid(tsk, PIDTYPE_PID);
>   tsk->pid = leader->pid;
> - attach_pid(tsk, PIDTYPE_PID, find_pid(tsk->pid));
> + attach_pid(tsk, PIDTYPE_PID, task_pid(leader));
>   transfer_pid(leader, tsk, PIDTYPE_PGID);
>   transfer_pid(leader, tsk, PIDTYPE_SID);
>   list_replace_rcu(&leader->tasks, &tsk->tasks);
> _____
> 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