Subject: [RFC][PATCH 09/14] Save leaders struct pid before detach_pid()
Posted by Sukadev Bhattiprolu on Wed, 21 Mar 2007 03:21:03 GMT
View Forum Message <> Reply to Message

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Subject: [RFC][PATCH 09/14] Save leaders struct pid before detach_pid()

Save the struct pid of the thread group leader before detaching
our pid. See comments in the code below for more details.

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
---
 fs/exec.c | 17 +++++++++++++++++-
 1 file changed, 16 insertions(+), 1 deletion(-)

Index: lx26-21-rc3-mm2/fs/exec.c
==========
--- lx26-21-rc3-mm2.orig/fs/exec.c 2007-03-20 15:55:52.000000000 -0700
+++ lx26-21-rc3-mm2/fs/exec.c 2007-03-20 17:17:38.000000000 -0700
@@ -578,6 +578,7 @@ static int de_thread(struct task_struct
  struct sighand_struct *newsighand, *oldsighand = tsk->sighand;
  spinlock_t *lock = &oldsighand->siglock;
  struct task_struct *leader = NULL;
+ struct pid *leader_pid;
  int count;

  /*
@@ -698,10 +699,24 @@ 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: Support for multiple pid namespaces implies that there
+   *   can be multiple processes with a given pid_t value,
+   *   and so, find_pid() needs/uses a notion of "current"
+   *   pid namespace to select the appropriate process to
+   *   return.
+   *
+   *   Since the current pid namespace of a process is stored
+   *   in its pid->pid_nrs list, if we detach/free our struct
+   *   pid, we would have no way of finding our current pid
+   *   namespace and so no way of finding the leader pid.
+   *
+   *   To work around, find the leader pid, before detaching
+   *   our pid from the task.
    */
+ leader_pid = find_pid(leader->pid);
   detach_pid(tsk, PIDTYPE_PID);
   tsk->pid = leader->pid;

```
-  attach_pid(tsk, PIDTYPE_PID,  find_pid(tsk->pid));
+  attach_pid(tsk, PIDTYPE_PID,  leader_pid);
   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: [RFC][PATCH 09/14] Save leaders struct pid before detach_pid()
Posted by ebiederm on Wed, 21 Mar 2007 07:31:38 GMT
View Forum Message <> Reply to Message

sukadev@us.ibm.com writes:

> From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
> Subject: [RFC][PATCH 09/14] Save leaders struct pid before detach_pid()
>
> Save the struct pid of the thread group leader before detaching
> our pid. See comments in the code below for more details.
>
> Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
> ---
> fs/exec.c | 17 +++++++++++++++-
> 1 file changed, 16 insertions(+), 1 deletion(-)
>
> Index: lx26-21-rc3-mm2/fs/exec.c
> ==
> --- lx26-21-rc3-mm2.orig/fs/exec.c 2007-03-20 15:55:52.000000000 -0700
> +++ lx26-21-rc3-mm2/fs/exec.c 2007-03-20 17:17:38.000000000 -0700
> @@ -578,6 +578,7 @@ static int de_thread(struct task_struct
>   struct sighand_struct *newsighand, *oldsighand = tsk->sighand;
>   spinlock_t *lock = &oldsighand->siglock;
>   struct task_struct *leader = NULL;
> + struct pid *leader_pid;
>   int count;
>
>   /*
> @@ -698,10 +699,24 @@ 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: Support for multiple pid namespaces implies that there
> + *   can be multiple processes with a given pid_t value,
> + *   and so, find_pid() needs/uses a notion of "current"
> + *   pid namespace to select the appropriate process to
```
```

> + * return.
> + *
> + * Since the current pid namespace of a process is stored
> + * in its pid->pid_nrs list, if we detach/free our struct
> + * pid, we would have no way of finding our current pid
> + * namespace and so no way of finding the leader pid.
> + *
> + * To work around, find the leader pid, before detaching
> + * our pid from the task.
>   */
> + leader_pid = find_pid(leader->pid);
>   detach_pid(tsk, PIDTYPE_PID);
>   tsk->pid = leader->pid;
> - attach_pid(tsk, PIDTYPE_PID, find_pid(tsk->pid));
> + attach_pid(tsk, PIDTYPE_PID, leader_pid);
How about just:
  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);

Eric

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