
Subject: [PATCH 03/17] Pid-NS(V3) Rename child_reaper function.
Posted by [Sukadev Bhattiprolu](#) on Sat, 16 Jun 2007 22:59:06 GMT
[View Forum Message](#) <> [Reply to Message](#)

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

Rename the child_reaper() function to task_child_reaper() to be similar to other task_* functions and to distinguish the function from 'struct pid_namespace.child_reaper'.

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>

fs/exec.c | 2 +-
include/linux/pid_namespace.h | 2 +-
kernel/exit.c | 4 +++
kernel/signal.c | 2 +-
4 files changed, 5 insertions(+), 5 deletions(-)

Index: lx26-22-rc4-mm2/include/linux/pid_namespace.h

=====

--- lx26-22-rc4-mm2.orig/include/linux/pid_namespace.h 2007-06-13 16:54:55.000000000 -0700
+++ lx26-22-rc4-mm2/include/linux/pid_namespace.h 2007-06-13 16:55:50.000000000 -0700
@@ -42,7 +42,7 @@ static inline struct pid_namespace *task
return tsk->nsproxy->pid_ns;
}

-static inline struct task_struct *child_reaper(struct task_struct *tsk)
+static inline struct task_struct *task_child_reaper(struct task_struct *tsk)
{
return init_pid_ns.child_reaper;
}

Index: lx26-22-rc4-mm2/fs/exec.c

=====

--- lx26-22-rc4-mm2.orig/fs/exec.c 2007-06-13 16:54:55.000000000 -0700
+++ lx26-22-rc4-mm2/fs/exec.c 2007-06-13 16:55:50.000000000 -0700
@@ -631,7 +631,7 @@ static int de_thread(struct task_struct
* Reparenting needs write_lock on tasklist_lock,
* so it is safe to do it under read_lock.
*/
- if (unlikely(tsk->group_leader == child_reaper(tsk)))
+ if (unlikely(tsk->group_leader == task_child_reaper(tsk)))
task_active_pid_ns(tsk)->child_reaper = tsk;

zap_other_threads(tsk);

Index: lx26-22-rc4-mm2/kernel/exit.c

=====

--- lx26-22-rc4-mm2.orig/kernel/exit.c 2007-06-13 16:54:55.000000000 -0700

+++ lx26-22-rc4-mm2/kernel/exit.c 2007-06-13 16:55:50.000000000 -0700

@@ -695,7 +695,7 @@ forget_original_parent(struct task_struc

```
do {
    reaper = next_thread(reaper);
    if (reaper == father) {
-   reaper = child_reaper(father);
+   reaper = task_child_reaper(father);
        break;
    }
} while (reaper->exit_state);
```

@@ -880,7 +880,7 @@ fastcall NORET_TYPE void do_exit(long co

```
panic("Aiee, killing interrupt handler!");
if (unlikely(!tsk->pid))
    panic("Attempted to kill the idle task!");
- if (unlikely(tsk == child_reaper(tsk))) {
+ if (unlikely(tsk == task_child_reaper(tsk))) {
    if (task_active_pid_ns(tsk) != &init_pid_ns)
        task_active_pid_ns(tsk)->child_reaper =
            init_pid_ns.child_reaper;
```

Index: lx26-22-rc4-mm2/kernel/signal.c

=====

--- lx26-22-rc4-mm2.orig/kernel/signal.c 2007-06-13 16:42:12.000000000 -0700

+++ lx26-22-rc4-mm2/kernel/signal.c 2007-06-13 16:55:50.000000000 -0700

@@ -1829,7 +1829,7 @@ relock:

```
* within that pid space. It can of course get signals from
* its parent pid space.
*/
- if (current == child_reaper(current))
+ if (current == task_child_reaper(current))
    continue;
```

```
    if (sig_kernel_stop(signr)) {
```

--

Containers mailing list

Containers@lists.linux-foundation.org

<https://lists.linux-foundation.org/mailman/listinfo/containers>
