
Subject: [RFC, PATCH] handle the multi-threaded init's exit() properly
Posted by [Oleg Nesterov](#) on Thu, 02 Aug 2007 21:20:09 GMT

[View Forum Message](#) <> [Reply to Message](#)

With or without this patch, multi-threaded init's are not fully supported, but do_exit() is completely wrong. This becomes a real problem when we support pid namespaces.

1. do_exit() panics when the main thread of /sbin/init exits. It should not until the whole thread group exits. Move the code below, under the "if (group_dead)" check.

Note: this means that forget_original_parent() can use an already dead child_reaper()'s task_struct. This is OK for /sbin/init because

- do_wait() from alive sub-thread still can reap a zombie, we iterate over all sub-thread's ->children lists
- do_notify_parent() will wakeup some alive sub-thread because it sends the group-wide signal

However, we should remove choose_new_parent()->BUG_ON(reaper->exit_state) for this.

2. We are playing games with ->nsproxy->pid_ns. This code is bogus today, and it has to be changed anyway when we really support pid namespaces, just remove it.

Signed-off-by: Oleg Nesterov <oleg@tv-sign.ru>

```
--- t/kernel/exit.c~ 2007-08-03 00:10:28.000000000 +0400
+++ t/kernel/exit.c 2007-08-03 01:12:18.000000000 +0400
@@ -604,11 +604,6 @@ static void exit_mm(struct task_struct *
static inline void
choose_new_parent(struct task_struct *p, struct task_struct *reaper)
{
- /*
- * Make sure we're not reparenting to ourselves and that
- * the parent is not a zombie.
- */
- BUG_ON(p == reaper || reaper->exit_state);
  p->real_parent = reaper;
}

@@ -895,6 +890,14 @@ static void check_stack_usage(void)
static inline void check_stack_usage(void) {}
#endif
```

```

+static inline void exit_child_reaper(struct task_struct *tsk)
+{
+ if (likely(tsk->group_leader != child_reaper(tsk)))
+ return;
+
+ panic("Attempted to kill init!");
+}
+
fastcall NORET_TYPE void do_exit(long code)
{
  struct task_struct *tsk = current;
@@ -908,13 +911,6 @@ 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 (tsk->nsproxy->pid_ns != &init_pid_ns)
- tsk->nsproxy->pid_ns->child_reaper = init_pid_ns.child_reaper;
- else
- panic("Attempted to kill init!");
- }
-

  if (unlikely(current->ptrace & PT_TRACE_EXIT)) {
    current->ptrace_message = code;
@@ -964,6 +960,7 @@ fastcall NORET_TYPE void do_exit(long co
  }
  group_dead = atomic_dec_and_test(&tsk->signal->live);
  if (group_dead) {
+ exit_child_reaper(tsk);
  hrtimer_cancel(&tsk->signal->real_timer);
  exit_itimers(tsk->signal);
  }

```

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