Subject: [PATCH 9/15] Move alloc_pid() after the namespace is cloned
Posted by Pavel Emelianov on Thu, 26 Jul 2007 14:52:33 GMT
View Forum Message <> Reply to Message

When we create new namespace we will need to allocate the struct pid,
that will have one extra struct upid in array, comparing to the parent.
Thus we need to know the new namespace (if any) in alloc_pid() to init
this struct upid properly, so move the alloc_pid() call lower in
copy_process().

This is a fix for Sukadev's patch that moved the alloc_pid() call from
do_fork() into copy_process().

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
---

 fork.c |   59 ++++++++++++++++++++++++++++++++++++++---------------------
 1 files changed, 38 insertions(+), 21 deletions(-)

diff -upr linux-2.6.23-rc1-mm1.orig/kernel/fork.c linux-2.6.23-rc1-mm1-7/kernel/fork.c
--- linux-2.6.23-rc1-mm1.orig/kernel/fork.c 2007-07-26 16:34:45.000000000 +0400
+++ linux-2.6.23-rc1-mm1-7/kernel/fork.c 2007-07-26 16:36:37.000000000 +0400
@@ -1028,16 +1029,9 @@ static struct task_struct *copy_process(
  if (p->binfmt && !try_module_get(p->binfmt->module))
   goto bad_fork_cleanup_put_domain;

- if (pid != &init_struct_pid) {
-  pid = alloc_pid();
-  if (!pid)
-   goto bad_fork_put_binfmt_module;
- }
-
  p->did_exec = 0;
  delayacct_tsk_init(p); /* Must remain after dup_task_struct() */
  copy_flags(clone_flags, p);
- p->pid = pid_nr(pid);
  INIT_LIST_HEAD(&p->children);
  INIT_LIST_HEAD(&p->sibling);
  p->vfork_done = NULL;
@@ -1112,10 +1106,6 @@ static struct task_struct *copy_process(
  p->blocked_on = NULL; /* not blocked yet */
 #endif

- p->tgid = p->pid;
- if (clone_flags & CLONE_THREAD)
-  p->tgid = current->tgid;
-
```

```
  if ((retval = security_task_alloc(p)))
   goto bad_fork_cleanup_policy;
  if ((retval = audit_alloc(p)))
@@ -1141,6 +1131,17 @@ static struct task_struct *copy_process(
  if (retval)
   goto bad_fork_cleanup_namespaces;

+ if (pid != &init_struct_pid) {
+  pid = alloc_pid(task_active_pid_ns(p));
+  if (!pid)
+   goto bad_fork_cleanup_namespaces;
+ }
+
+ p->pid = pid_nr(pid);
+ p->tgid = p->pid;
+ if (clone_flags & CLONE_THREAD)
+  p->tgid = current->tgid;
+
  p->set_child_tid = (clone_flags & CLONE_CHILD_SETTID) ? child_tidptr : NULL;
  /*
   * Clear TID on mm_release()?
@@ -1237,7 +1242,7 @@ static struct task_struct *copy_process(
   spin_unlock(&current->sighand->siglock);
   write_unlock_irq(&tasklist_lock);
   retval = -ERESTARTNOINTR;
-  goto bad_fork_cleanup_namespaces;
+  goto bad_fork_free_pid;
  }

  if (clone_flags & CLONE_THREAD) {
@@ -1266,11 +1271,22 @@ static struct task_struct *copy_process(
   __ptrace_link(p, current->parent);

   if (thread_group_leader(p)) {
-   p->signal->tty = current->signal->tty;
-   p->signal->pgrp = task_pgrp_nr(current);
-   set_task_session(p, task_session_nr(current));
-   attach_pid(p, PIDTYPE_PGID, task_pgrp(current));
-   attach_pid(p, PIDTYPE_SID, task_session(current));
+   if (clone_flags & CLONE_NEWPID) {
+    p->nsproxy->pid_ns->child_reaper = p;
+    p->signal->tty = NULL;
+    p->signal->pgrp = p->pid;
+    set_task_session(p, p->pid);
+    attach_pid(p, PIDTYPE_PGID, pid);
+    attach_pid(p, PIDTYPE_SID, pid);
+   } else {
+    p->signal->tty = current->signal->tty;
```

```
+    p->signal->pgrp = task_pgrp_nr(current);
+    set_task_session(p, task_session_nr(current));
+    attach_pid(p, PIDTYPE_PGID,
+      task_pgrp(current));
+    attach_pid(p, PIDTYPE_SID,
+      task_session(current));
+   }

    list_add_tail_rcu(&p->tasks, &init_task.tasks);
    __get_cpu_var(process_counts)++;
@@ -1288,6 +1304,9 @@ static struct task_struct *copy_process(
  container_post_fork(p);
  return p;

+bad_fork_free_pid:
+ if (pid != &init_struct_pid)
+  free_pid(pid);
 bad_fork_cleanup_namespaces:
  exit_task_namespaces(p);
 bad_fork_cleanup_keys:
@@ -1322,9 +1341,6 @@ bad_fork_cleanup_container:
 #endif
  container_exit(p, container_callbacks_done);
  delayacct_tsk_free(p);
- if (pid != &init_struct_pid)
-  free_pid(pid);
-bad_fork_put_binfmt_module:
  if (p->binfmt)
   module_put(p->binfmt->module);
 bad_fork_cleanup_put_domain:
@@ -1406,7 +1422,13 @@ long do_fork(unsigned long clone_flags,
  if (!IS_ERR(p)) {
   struct completion vfork;

-  nr = pid_nr(task_pid(p));
+  /*
+   * this is enough to call pid_nr_ns here, but this if
+   * improves optimisation of regular fork()
+   */
+  nr = (clone_flags & CLONE_NEWPID) ?
+   task_pid_nr_ns(p, current->nsproxy->pid_ns) :
+    task_pid_vnr(p);

   if (clone_flags & CLONE_VFORK) {
    p->vfork_done = &vfork;
```