Subject: [PATCH] Remove find_attach_pid()
Posted by Sukadev Bhattiprolu on Fri, 12 Jan 2007 03:27:43 GMT
View Forum Message <> Reply to Message

As Eric Biederman pointed out, find_attach_pid() is not really necessary.

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

Remove find_attach_pid() interface and have callers of attach_pid() pass in
a struct pid.

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Cc: Cedric Le Goater <clg@fr.ibm.com>
Cc: Dave Hansen <haveblue@us.ibm.com>
Cc: Serge Hallyn <serue@us.ibm.com>
Cc: Eric W. Biederman <ebiederm@xmission.com>
Cc: containers@lists.osdl.org

---
 fs/exec.c | 2 +-
 include/linux/pid.h | 11 ++---------
 kernel/exit.c | 4 ++--
 kernel/fork.c | 13 ++++++++-----
 kernel/pid.c | 6 ++++++
 kernel/sys.c | 2 +-
 6 files changed, 20 insertions(+), 18 deletions(-)

Index: lx26-20-rc2-mm1/fs/exec.c
===
--- lx26-20-rc2-mm1.orig/fs/exec.c 2007-01-10 21:42:43.000000000 -0800
+++ lx26-20-rc2-mm1/fs/exec.c 2007-01-11 19:06:00.260386560 -0800
@@ -701,7 +701,7 @@ static int de_thread(struct task_struct
     */
    detach_pid(tsk, PIDTYPE_PID);
    tsk->pid = leader->pid;
-   find_attach_pid(tsk, PIDTYPE_PID, tsk->pid);
+   attach_pid(tsk, PIDTYPE_PID, find_pid(tsk->pid));
    transfer_pid(leader, tsk, PIDTYPE_PGID);
    transfer_pid(leader, tsk, PIDTYPE_SID);
    list_replace_rcu(&leader->tasks, &tsk->tasks);
Index: lx26-20-rc2-mm1/kernel/exit.c
===
--- lx26-20-rc2-mm1.orig/kernel/exit.c 2007-01-10 21:43:09.000000000 -0800
+++ lx26-20-rc2-mm1/kernel/exit.c 2007-01-11 18:36:34.000000000 -0800
@@ -301,12 +301,12 @@ void __set_special_pids(pid_t session, p
   if (process_session(curr) != session) {
    detach_pid(curr, PIDTYPE_SID);

```
   set_signal_session(curr->signal, session);
-  find_attach_pid(curr, PIDTYPE_SID, session);
+  attach_pid(curr, PIDTYPE_SID, find_pid(session));
   }
   if (process_group(curr) != pgrp) {
    detach_pid(curr, PIDTYPE_PGID);
    curr->signal->pgrp = pgrp;
-  find_attach_pid(curr, PIDTYPE_PGID, pgrp);
+  attach_pid(curr, PIDTYPE_PGID, find_pid(pgrp));
   }
  }

Index: lx26-20-rc2-mm1/kernel/fork.c
===================================================================
--- lx26-20-rc2-mm1.orig/kernel/fork.c 2007-01-11 18:29:33.000000000 -0800
+++ lx26-20-rc2-mm1/kernel/fork.c 2007-01-11 18:39:09.000000000 -0800
@@ -1245,16 +1245,19 @@ static struct task_struct *copy_process(
    __ptrace_link(p, current->parent);

    if (thread_group_leader(p)) {
+    pid_t pgid = process_group(current);
+    pid_t sid = process_session(current);
+
     p->signal->tty = current->signal->tty;
-    p->signal->pgrp = process_group(current);
-    set_signal_session(p->signal, process_session(current));
-    find_attach_pid(p, PIDTYPE_PGID, process_group(p));
-    find_attach_pid(p, PIDTYPE_SID, process_session(p));
+    p->signal->pgrp = pgid;
+    set_signal_session(p->signal, sid);
+    attach_pid(p, PIDTYPE_PGID, find_pid(pgid));
+    attach_pid(p, PIDTYPE_SID, find_pid(sid));

     list_add_tail_rcu(&p->tasks, &init_task.tasks);
     __get_cpu_var(process_counts)++;
    }
-  find_attach_pid(p, PIDTYPE_PID, p->pid);
+  attach_pid(p, PIDTYPE_PID, find_pid(p->pid));
    nr_threads++;
   }

Index: lx26-20-rc2-mm1/include/linux/pid.h
===================================================================
--- lx26-20-rc2-mm1.orig/include/linux/pid.h 2007-01-11 07:18:03.000000000 -0800
+++ lx26-20-rc2-mm1/include/linux/pid.h 2007-01-11 18:41:38.861552952 -0800
@@ -84,18 +84,11 @@ extern struct pid *find_get_pid(int nr);
 extern struct pid *find_ge_pid(int nr);
```

```
 /*
- * attach_pid(), find_attach_pid() and detach_pid() must be called with the
- * tasklist_lock write-held.
+ * attach_pid() and detach_pid() must be called with the tasklist_lock
+ * write-held.
  */
 extern int FASTCALL(attach_pid(struct task_struct *task, enum pid_type type,
     struct pid *pid));
-
-static inline int find_attach_pid(struct task_struct *task, enum pid_type type,
-    int nr)
-{
- return attach_pid(task, type, find_pid(nr));
-}
-
 extern void FASTCALL(detach_pid(struct task_struct *task, enum pid_type));
 extern void FASTCALL(transfer_pid(struct task_struct *old,
       struct task_struct *new, enum pid_type));
Index: lx26-20-rc2-mm1/kernel/sys.c
===================================================================
--- lx26-20-rc2-mm1.orig/kernel/sys.c 2007-01-10 21:45:01.000000000 -0800
+++ lx26-20-rc2-mm1/kernel/sys.c 2007-01-11 18:40:01.000000000 -0800
@@ -1480,7 +1480,7 @@ asmlinkage long sys_setpgid(pid_t pid, p
  if (process_group(p) != pgid) {
   detach_pid(p, PIDTYPE_PGID);
   p->signal->pgrp = pgid;
-  find_attach_pid(p, PIDTYPE_PGID, pgid);
+  attach_pid(p, PIDTYPE_PGID, find_pid(pgid));
  }

  err = 0;
Index: lx26-20-rc2-mm1/kernel/pid.c
===================================================================
--- lx26-20-rc2-mm1.orig/kernel/pid.c 2007-01-11 07:18:03.000000000 -0800
+++ lx26-20-rc2-mm1/kernel/pid.c 2007-01-11 19:00:09.997634592 -0800
@@ -247,6 +247,9 @@ struct pid * fastcall find_pid(int nr)
 }
 EXPORT_SYMBOL_GPL(find_pid);

+/*
+ * attach_pid() must be called with the tasklist_lock write-held.
+ */
 int fastcall attach_pid(struct task_struct *task, enum pid_type type,
     struct pid *pid)
 {
@@ -259,6 +262,9 @@ int fastcall attach_pid(struct task_stru
  return 0;
 }
```

```
+/*
+ * detach_pid() must be called with the tasklist_lock write-held.
+ */
 void fastcall detach_pid(struct task_struct *task, enum pid_type type)
 {
  struct pid_link *link;
```

_____

Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers