Subject: [PATCH 4/6] Use pid_to_nr() in process info functions
Posted by Sukadev Bhattiprolu on Fri, 06 Jul 2007 05:55:12 GMT
View Forum Message <> Reply to Message

Subject: [PATCH 4/6] Use pid_to_nr() in process info functions

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

Use pid_to_nr() function in getppid(), getpgid() and getsid() functions
so they return the correct pid_t for processes in multiple pid namespaces.

Note: We don't need pid_to_nr() in getpid() because the process always
"sees itself" as being in its active pid namespace. Using pid_to_nr()
in getpid() would unnecessarily hurt its performance.

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
---
include/linux/sched.h | 24 ++++++++++++++++++++++
kernel/sys.c | 10 +++++-----
kernel/timer.c | 2 +-
3 files changed, 30 insertions(+), 6 deletions(-)

Index: lx26-22-rc6-mm1/include/linux/sched.h
===
--- lx26-22-rc6-mm1.orig/include/linux/sched.h 2007-07-05 18:53:42.000000000 -0700
+++ lx26-22-rc6-mm1/include/linux/sched.h 2007-07-05 18:57:34.000000000 -0700
@@ -1200,6 +1200,30 @@ static inline struct pid *task_tgid(stru
return task->group_leader->pids[PIDTYPE_PID].pid;
}

+/* NOTE: Caller must hold rcu_readlock() */
+static inline struct pid *task_parent_tgid(struct task_struct *task)
+{
+ return task_tgid(rcu_dereference(task->real_parent));
+}
+
+/* NOTE: Caller must hold rcu_readlock() */
+static inline struct pid *task_parent_pid(struct task_struct *task)
+{
+ return task_pid(rcu_dereference(task->real_parent));
+}
+
+/* NOTE: Caller must hold rcu_readlock() */
+static inline struct pid *task_tracer_tgid(struct task_struct *task)
+{
+ return task_tgid(rcu_dereference(task->parent));
+}
+

```
+/* NOTE: Caller must hold rcu_readlock() */
+static inline struct pid *task_tracer_pid(struct task_struct *task)
+{
+ return task_pid(rcu_dereference(task->parent));
+}
+
 static inline struct pid *task_pgrp(struct task_struct *task)
 {
  return task->group_leader->pids[PIDTYPE_PGID].pid;
Index: lx26-22-rc6-mm1/kernel/timer.c
===================================================================
--- lx26-22-rc6-mm1.orig/kernel/timer.c 2007-07-05 18:53:42.000000000 -0700
+++ lx26-22-rc6-mm1/kernel/timer.c 2007-07-05 18:57:34.000000000 -0700
@@ -971,7 +971,7 @@ asmlinkage long sys_getppid(void)
  int pid;

  rcu_read_lock();
- pid = rcu_dereference(current->real_parent)->tgid;
+ pid = pid_to_nr(task_parent_tgid(current));
  rcu_read_unlock();

  return pid;
Index: lx26-22-rc6-mm1/kernel/sys.c
===================================================================
--- lx26-22-rc6-mm1.orig/kernel/sys.c 2007-07-05 18:53:42.000000000 -0700
+++ lx26-22-rc6-mm1/kernel/sys.c 2007-07-05 18:57:34.000000000 -0700
@@ -1503,7 +1503,7 @@ out:
 asmlinkage long sys_getpgid(pid_t pid)
 {
  if (!pid)
-  return process_group(current);
+  return pid_to_nr(task_pgrp(current));
  else {
   int retval;
   struct task_struct *p;
@@ -1515,7 +1515,7 @@ asmlinkage long sys_getpgid(pid_t pid)
   if (p) {
    retval = security_task_getpgid(p);
    if (!retval)
-    retval = process_group(p);
+    retval = pid_to_nr(task_pgrp(p));
   }
   read_unlock(&tasklist_lock);
   return retval;
@@ -1527,7 +1527,7 @@ asmlinkage long sys_getpgid(pid_t pid)
 asmlinkage long sys_getpgrp(void)
 {
  /* SMP - assuming writes are word atomic this is fine */
```

```
- return process_group(current);
+ return pid_to_nr(task_pgrp(current));
 }

 #endif
@@ -1535,7 +1535,7 @@ asmlinkage long sys_getpgrp(void)
 asmlinkage long sys_getsid(pid_t pid)
 {
  if (!pid)
-  return process_session(current);
+  return pid_to_nr(task_session(current));
  else {
   int retval;
   struct task_struct *p;
@@ -1547,7 +1547,7 @@ asmlinkage long sys_getsid(pid_t pid)
   if (p) {
    retval = security_task_getsid(p);
    if (!retval)
-    retval = process_session(p);
+    retval = pid_to_nr(task_session(p));
   }
   read_unlock(&tasklist_lock);
   return retval;
```