Subject: [PATCH 2/3] Pid ns helpers for signals
Posted by Sukadev Bhattiprolu on Tue, 11 Sep 2007 04:11:21 GMT
View Forum Message <> Reply to Message

Define some helper functions that will be used to implement signal semantics
with multiple pid namespaces.

 is_current_in_ancestor_pid_ns(task)

  TRUE iff active pid namespace of 'current' is an ancestor of
  active pid namespace of @task.

 is_current_in_same_or_ancestor_pid_ns(task)

  TRUE iff active pid namespace of 'current' is either same as
  or an ancestor of active pid namespace of @task.

 pid_ns_equal(tsk)
  TRUE if active pid ns of @tsk is same as active pid ns of
  'current'.

Changelog: [Oleg Nesterov]: Renamed helpers. Dropped reference to pid and
      pid-namespace since they are stable for current
      callers.

---
 include/linux/pid.h | 12 +++++++++
 kernel/pid.c | 63 ++
 2 files changed, 75 insertions(+)

Index: 2.6.23-rc4-mm1/include/linux/pid.h
===
--- 2.6.23-rc4-mm1.orig/include/linux/pid.h 2007-09-07 19:18:34.000000000 -0700
+++ 2.6.23-rc4-mm1/include/linux/pid.h 2007-09-07 19:18:42.000000000 -0700
@@ -124,6 +124,18 @@ extern struct pid *alloc_pid(struct pid_
 extern void FASTCALL(free_pid(struct pid *pid));
 extern void zap_pid_ns_processes(struct pid_namespace *pid_ns);

+static inline struct pid_namespace *pid_active_ns(struct pid *pid)
+{
+ if (!pid)
+ return NULL;
+
+ return pid->numbers[pid->level].ns;
+}
+
+extern int pid_ns_equal(struct task_struct *tsk);
+extern int is_current_in_ancestor_pid_ns(struct task_struct *tsk);

```
+extern int is_current_in_same_or_ancestor_pid_ns(struct task_struct *tsk);
+
 /*
  * the helpers to get the pid's id seen from different namespaces
  *
Index: 2.6.23-rc4-mm1/kernel/pid.c
===================================================================
--- 2.6.23-rc4-mm1.orig/kernel/pid.c 2007-09-07 19:18:34.000000000 -0700
+++ 2.6.23-rc4-mm1/kernel/pid.c 2007-09-10 18:35:51.000000000 -0700
@@ -199,6 +199,69 @@ static int next_pidmap(struct pid_namesp
  return -1;
 }

+/*
+ * Return TRUE if the active pid namespace of @tsk is same as active
+ * pid namespace of 'current'.
+ */
+int pid_ns_equal(struct task_struct *tsk)
+{
+ struct pid_namespace *my_ns = pid_active_ns(task_pid(current));
+ struct pid_namespace *tsk_ns = pid_active_ns(task_pid(tsk));
+
+ return my_ns == tsk_ns;
+}
+
+/*
+ * Return TRUE if pid namespace @ns1 is an ancestor of pid namespace @ns2.
+ * Return FALSE otherwise.
+ *
+ * Note: Callers must ensure @ns1 and @ns2 are stable.
+ */
+static int ancestor_pid_ns(struct pid_namespace *ns1, struct pid_namespace *ns2)
+{
+ int i;
+ struct pid_namespace *tmp;
+
+ if (ns1 == NULL || ns2 == NULL)
+  return 0;
+
+ if (ns1->level >= ns2->level)
+  return 0;
+
+ tmp = ns2->parent;
+ for (i = tmp->level; i >= ns1->level; i--) {
+  if (tmp == ns1)
+   return 1;
+  tmp = tmp->parent;
+ }
```

```
+
+ return 0;
+}
+
+/*
+ * Return TRUE if active pid namespace of 'current' is an ancestor of
+ * pid namespace of @tsk. Return FALSE otherwise.
+ */
+int is_current_in_ancestor_pid_ns(struct task_struct *tsk)
+{
+ struct pid_namespace *my_ns = pid_active_ns(task_pid(current));
+ struct pid_namespace *tsk_ns = pid_active_ns(task_pid(tsk));
+
+ return ancestor_pid_ns(my_ns, tsk_ns);
+}
+
+/*
+ * Return TRUE if active pid namespace of 'current' is either same as
+ * or an ancestor of active pid namespace of @tsk.
+ */
+int is_current_in_same_or_ancestor_pid_ns(struct task_struct *tsk)
+{
+ struct pid_namespace *my_ns = pid_active_ns(task_pid(current));
+ struct pid_namespace *tsk_ns = pid_active_ns(task_pid(tsk));
+
+ return my_ns == tsk_ns || ancestor_pid_ns(my_ns, tsk_ns);
+}
+
 fastcall void put_pid(struct pid *pid)
 {
  struct pid_namespace *ns;
```