
Subject: Re: [PATCH 12/15] Miscellaneous stuff for pid namespaces
Posted by [Sukadev Bhattiprolu](#) on Fri, 27 Jul 2007 06:22:13 GMT
[View Forum Message](#) <> [Reply to Message](#)

Pavel Emelianov [xemul@openvz.org] wrote:

| This includes
| * the headers dependency fix
| * task_child_reaper() correct declaration
| * fixes for is_global_init() and is_container_init()
|
| Maybe this should come before all the other stuff...
|
| Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

| ---

| include/linux/pid_namespace.h | 4 ++--
| include/linux/sched.h | 4 ++--
| kernel/pid.c | 16 ++++++++-----
| 3 files changed, 17 insertions(+), 7 deletions(-)
|
| diff -upr linux-2.6.23-rc1-mm1.orig/include/linux/pid_namespace.h
| linux-2.6.23-rc1-mm1-7/include/linux/pid_namespace.h
| --- linux-2.6.23-rc1-mm1.orig/include/linux/pid_namespace.h 2007-07-26
| 16:34:45.000000000 +0400
| +++ linux-2.6.23-rc1-mm1-7/include/linux/pid_namespace.h 2007-07-26
| 16:36:36.000000000 +0400
| @@ -4,7 +4,6 @@
| #include <linux/sched.h>
| #include <linux/mm.h>
| #include <linux/threads.h>
| -#include <linux/pid.h>
| #include <linux/nsproxy.h>
| #include <linux/kref.h>
|
| @@ -46,7 +53,8 @@ static inline struct pid_namespace *task
|
| static inline struct task_struct *task_child_reaper(struct task_struct *tsk)
| {
| - return init_pid_ns.child_reaper;
| + BUG_ON(tsk != current);
| + return tsk->nsproxy->pid_ns->child_reaper;
| }
|
| #endif /* _LINUX_PID_NS_H */
| diff -upr linux-2.6.23-rc1-mm1.orig/include/linux/sched.h
| linux-2.6.23-rc1-mm1-7/include/linux/sched.h
| --- linux-2.6.23-rc1-mm1.orig/include/linux/sched.h 2007-07-26

```
| 16:34:45.000000000 +0400
| +++ linux-2.6.23-rc1-mm1-7/include/linux/sched.h 2007-07-26
| 16:36:37.000000000 +0400
| @@ -1277,13 +1366,13 @@ static inline int pid_alive(struct task_
| *
| * TODO: We should inline this function after some cleanups in
| pid_namespace.h
| */
```

We can now remove this TODO.

```
| -extern int is_global_init(struct task_struct *tsk);
| +extern int is_container_init(struct task_struct *tsk);
|
| /*
| * is_container_init:
| * check whether in the task is init in its own pid namespace.
| */
```

The function header describes `is_container_init()` but the function is actually `is_global_init()`. The opp is true for the function heaer of `is_container_init()` above.

```
| -static inline int is_container_init(struct task_struct *tsk)
| +static inline int is_global_init(struct task_struct *tsk)
| {
|     return tsk->pid == 1;
| }
| diff -upr linux-2.6.23-rc1-mm1.orig/kernel/pid.c
| linux-2.6.23-rc1-mm1-7/kernel/pid.c
| --- linux-2.6.23-rc1-mm1.orig/kernel/pid.c 2007-07-26
| 16:34:45.000000000 +0400
| +++ linux-2.6.23-rc1-mm1-7/kernel/pid.c 2007-07-26
| 16:36:37.000000000 +0400
| @@ -60,11 +62,21 @@ static inline int mk_pid(struct pid_name
| };
| EXPORT_SYMBOL(init_pid_ns);
|
| -int is_global_init(struct task_struct *tsk)
| +int is_container_init(struct task_struct *tsk)
| {
|     - return tsk == init_pid_ns.child_reaper;
|     + int ret;
```

Initialize `ret = 0` here would save a line of code below :-)

```
| + struct pid *pid;
| +
```

```
| + ret = 0;
| + rcu_read_lock();
| + pid = task_pid(tsk);
| + if (pid != NULL && pid->numbers[pid->level].nr == 1)
| + ret = 1;
| + rcu_read_unlock();
| +
| + return ret;
| }
| -EXPORT_SYMBOL(is_global_init);
| +EXPORT_SYMBOL(is_container_init);
|
| /*
|  * Note: disable interrupts while the pidmap_lock is held as an
```
