
Subject: Re: [PATCH 1/6] Define and use task_active_pid_ns() wrapper
Posted by [Pavel Emelianov](#) on Fri, 13 Jul 2007 05:19:44 GMT
[View Forum Message](#) <> [Reply to Message](#)

sukadev@us.ibm.com wrote:

> Subject: [PATCH 1/6] Define and use task_active_pid_ns() wrapper
>
> From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
>
> With multiple pid namespaces, a process is known by some pid_t in
> every ancestor pid namespace. Every time the process forks, the
> child process also gets a pid_t in every ancestor pid namespace.
>
> While a process is visible in >=1 pid namespaces, it can see pid_t's
> in only one pid namespace. We call this pid namespace it's "active
> pid namespace", and it is always the youngest pid namespace in which
> the process is known.
>
> This patch defines and uses a wrapper to find the active pid namespace
> of a process. The implementation of the wrapper will be changed in
> when support for multiple pid namespaces are added.
>
> Changelog:
> 2.6.22-rc4-mm2-pidns1:
> - [Pavel Emelianov, Alexey Dobriyan] Back out the change to use
> task_active_pid_ns() in child_reaper() since task->nsproxy
> can be NULL during task exit (so child_reaper() continues to
> use init_pid_ns).
>
> to implement child_reaper() since init_pid_ns.child_reaper to
> implement child_reaper() since task->nsproxy can be NULL during exit.
>
> 2.6.21-rc6-mm1:
> - Rename task_pid_ns() to task_active_pid_ns() to reflect that a
> process can have multiple pid namespaces.
>
> Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>

Acked-by: Pavel Emelianov <xemul@openvz.org>

But I'm not sure Andrew will like the construction like

some_function()->xxx;

I used to do so in memory controller, but he asked to fix...

> ---
> fs/exec.c | 2 +-
>

```

> fs/proc/proc_misc.c      | 3 ++-
> include/linux/pid_namespace.h | 5 +++++
> kernel/exit.c            | 5 +++--
> kernel/nsproxy.c         | 2 +-
> kernel/pid.c             | 4 ++--
> 6 files changed, 14 insertions(+), 7 deletions(-)
>
> Index: lx26-22-rc6-mm1/include/linux/pid_namespace.h
> =====
> --- lx26-22-rc6-mm1.orig/include/linux/pid_namespace.h 2007-07-05 18:53:43.000000000 -0700
> +++ lx26-22-rc6-mm1/include/linux/pid_namespace.h 2007-07-05 18:53:48.000000000 -0700
> @@ -37,6 +37,11 @@ static inline void put_pid_ns(struct pid
>  kref_put(&ns->kref, free_pid_ns);
> }
>
> +static inline struct pid_namespace *task_active_pid_ns(struct task_struct *tsk)
> +{
> + return tsk->nsproxy->pid_ns;
> +}
> +
> static inline struct task_struct *child_reaper(struct task_struct *tsk)
> {
> return init_pid_ns.child_reaper;
> Index: lx26-22-rc6-mm1/fs/exec.c
> =====
> --- lx26-22-rc6-mm1.orig/fs/exec.c 2007-07-05 18:53:43.000000000 -0700
> +++ lx26-22-rc6-mm1/fs/exec.c 2007-07-05 18:53:48.000000000 -0700
> @@ -827,7 +827,7 @@ static int de_thread(struct task_struct
>  * so it is safe to do it under read_lock.
>  */
>  if (unlikely(tsk->group_leader == child_reaper(tsk)))
> - tsk->nsproxy->pid_ns->child_reaper = tsk;
> + task_active_pid_ns(tsk)->child_reaper = tsk;
>
> zap_other_threads(tsk);
> read_unlock(&tasklist_lock);
> Index: lx26-22-rc6-mm1/fs/proc/proc_misc.c
> =====
> --- lx26-22-rc6-mm1.orig/fs/proc/proc_misc.c 2007-07-05 18:53:43.000000000 -0700
> +++ lx26-22-rc6-mm1/fs/proc/proc_misc.c 2007-07-05 18:53:48.000000000 -0700
> @@ -94,7 +94,8 @@ static int loadavg_read_proc(char *page,
>  LOAD_INT(a), LOAD_FRAC(a),
>  LOAD_INT(b), LOAD_FRAC(b),
>  LOAD_INT(c), LOAD_FRAC(c),
> - nr_running(), nr_threads, current->nsproxy->pid_ns->last_pid);
> + nr_running(), nr_threads,
> + task_active_pid_ns(current)->last_pid);
> return proc_calc_metrics(page, start, off, count, eof, len);

```

```

> }
>
> Index: lx26-22-rc6-mm1/kernel/exit.c
> =====
> --- lx26-22-rc6-mm1.orig/kernel/exit.c 2007-07-05 18:53:43.000000000 -0700
> +++ lx26-22-rc6-mm1/kernel/exit.c 2007-07-05 18:53:48.000000000 -0700
> @@ -909,8 +909,9 @@ fastcall NORET_TYPE void do_exit(long co
>  if (unlikely(!tsk->pid))
>  {
>      panic("Attempted to kill the idle task!");
>      if (unlikely(tsk == child_reaper(tsk))) {
>  - if (tsk->nsproxy->pid_ns != &init_pid_ns)
>  - tsk->nsproxy->pid_ns->child_reaper = init_pid_ns.child_reaper;
>  + if (task_active_pid_ns(tsk) != &init_pid_ns)
>  + task_active_pid_ns(tsk)->child_reaper =
>  + init_pid_ns.child_reaper;
>  else
>      panic("Attempted to kill init!");
>  }
> Index: lx26-22-rc6-mm1/kernel/pid.c
> =====
> --- lx26-22-rc6-mm1.orig/kernel/pid.c 2007-07-05 18:53:43.000000000 -0700
> +++ lx26-22-rc6-mm1/kernel/pid.c 2007-07-05 18:53:48.000000000 -0700
> @@ -213,7 +213,7 @@ struct pid *alloc_pid(void)
>  if (!pid)
>      goto out;
>
>  - nr = alloc_pidmap(current->nsproxy->pid_ns);
>  + nr = alloc_pidmap(task_active_pid_ns(current));
>  if (nr < 0)
>      goto out_free;
>
> @@ -358,7 +358,7 @@ struct pid *find_ge_pid(int nr)
>  pid = find_pid(nr);
>  if (pid)
>      break;
>  - nr = next_pidmap(current->nsproxy->pid_ns, nr);
>  + nr = next_pidmap(task_active_pid_ns(current), nr);
>  } while (nr > 0);
>
>  return pid;
> Index: lx26-22-rc6-mm1/kernel/nsproxy.c
> =====
> --- lx26-22-rc6-mm1.orig/kernel/nsproxy.c 2007-07-05 18:53:42.000000000 -0700
> +++ lx26-22-rc6-mm1/kernel/nsproxy.c 2007-07-05 18:54:07.000000000 -0700
> @@ -86,7 +86,7 @@ static struct nsproxy *create_new_namesp
>  goto out_ipc;
>  }
>

```

```
> - new_nsp->pid_ns = copy_pid_ns(flags, tsk->nsproxy->pid_ns);  
> + new_nsp->pid_ns = copy_pid_ns(flags, task_active_pid_ns(tsk));  
> if (IS_ERR(new_nsp->pid_ns)) {  
>     err = PTR_ERR(new_nsp->pid_ns);  
>     goto out_pid;  
>  
_____  
> Containers mailing list  
> Containers@lists.linux-foundation.org  
> https://lists.linux-foundation.org/mailman/listinfo/containers  
>
```

```
_____  
Containers mailing list  
Containers@lists.linux-foundation.org  
https://lists.linux-foundation.org/mailman/listinfo/containers
```
