
Subject: Re: [PATCH 10/15] Make each namespace has its own proc tree

Posted by [Oleg Nesterov](#) on Sun, 29 Jul 2007 15:56:50 GMT

[View Forum Message](#) <> [Reply to Message](#)

On 07/26, Pavel Emelyanov wrote:

```
>
> static int proc_get_sb(struct file_system_type *fs_type,
> int flags, const char *dev_name, void *data, struct vfsmount *mnt)
> {
> [...snip...]
>
> + if (!sb->s_root) {
> + sb->s_flags = flags;
> + err = proc_fill_super(sb);
> + if (err) {
> + up_write(&sb->s_umount);
> + deactivate_super(sb);
> + return err;
> + }
> +
> + ei = PROC_I(sb->s_root->d_inode);
> + if (!ei->pid) {
> + rcu_read_lock();
> + ei->pid = get_pid(find_pid_ns(1, ns));
```

Where do we "put" this pid?

```
> void release_task(struct task_struct * p)
> {
> + struct pid *pid;
> struct task_struct *leader;
> int zap_leader;
> repeat:
> @@ -160,6 +161,20 @@ repeat:
> write_lock_irq(&tasklist_lock);
> ptrace_unlink(p);
> BUG_ON(!list_empty(&p->ptrace_list) ||
> !list_empty(&p->ptrace_children));
> + /*
> + * we have to keep this pid till proc_flush_task() to make
> + * it possible to flush all dentries holding it. pid will
> + * be put ibidem
> + *
> + * however if the pid belongs to init namespace only, we can
> + * optimize this out
> + */
> + pid = task_pid(p);
> + if (pid->level != 0)
```

```

> + get_pid(pid);
> + else
> + pid = NULL;
> +
> __exit_signal(p);
>
> /*
> @@ -184,7 +199,8 @@ repeat:
> }
>
> write_unlock_irq(&tasklist_lock);
> - proc_flush_task(p, NULL);
> + proc_flush_task(p, pid);
> + put_pid(pid);

```

Oh, this is not nice...

Look, `proc_flush_task()` doesn't need `pid`, it can use `active_pid_ns` to iterate the `->parent` chain and do `proc_flush_task_mnt()`, and it can check `"ns->child_reaper == current"` for `mntput()`, yes?

So, can't we do instead

```

release_task()
{
    struct pid_namespace *my_ns = ...; // get it from pid;

    ...

    write_unlock_irq(&tasklist_lock);

    // __exit_signal()->...->put_pid() drops the reference,
    // but this ns should still be valid because /proc itself
    // holds a reference to it, even if we are /sbin/init.
    proc_flush_task(p, my_ns);
    ...
}

```

No?

Oleg.