Subject: Re: [PATCH] cgroup: fix a race condition in manipulating tsk->cg_list
Posted by Li Zefan on Thu, 17 Apr 2008 05:04:47 GMT
View Forum Message <> Reply to Message

Paul Menage wrote:

```
> On Wed, Apr 16, 2008 at 8:37 PM, Li Zefan <lizf@cn.fujitsu.com> wrote:
>> When I ran a test program to fork mass processes and at the same time
>>  'cat /cgroup/tasks', I got the following oops:
>>
>>  ------------[ cut here ]------------
>>  kernel BUG at lib/list_debug.c:72!
>>  invalid opcode: 0000 [#1] SMP
>>  Pid: 4178, comm: a.out Not tainted (2.6.25-rc9 #72)
>>  ...
>>  Call Trace:
>>   [<c044a5f9>] ? cgroup_exit+0x55/0x94
>>   [<c0427acf>] ? do_exit+0x217/0x5ba
>>   [<c0427ed7>] ? do_group_exit+0.65/0x7c
>>   [<c0427efd>] ? sys_exit_group+0xf/0x11
>>   [<c0404842>] ? syscall_call+0x7/0xb
>>   [<c05e0000>] ? init_cyrix+0x2fa/0x479
>>  ...
>>  EIP: [<c04df671>] list_del+0x35/0x53 SS:ESP 0068:ebc7df4
>>  ---[ end trace caffb7332252612b ]---
>>  Fixing recursive fault but reboot is needed!
>>
>>  After digging into the code and debugging, I finlly found out a race
>>  situation:
>>                                 do_exit()
>>                                   ->cgroup_exit()
>>                                     ->if (!list_empty(&tsk->cg_list))
>>                                         list_del(&tsk->cg_list);
>>
>>  cgroup_iter_start()
>>   ->cgroup_enable_task_cg_list()
>>     ->list_add(&tsk->cg_list, ..);
>>
>>  In this case the list won't be deleted though the process has exited.
>>
>>  We got two bug reports in the past, which seem to be the same bug as
>>  this one:
>>         http://lkml.org/lkml/2008/3/5/332
>>         http://lkml.org/lkml/2007/10/17/224
>
> Yes, that looks like it could be the same one - great. But this
> corruption can only be triggered the first time you cat a tasks file
> after a reboot, right? That would partly explain why it was hard to
> reproduce (at least, I had trouble).
```

>

Right. I was lucky to trigger this and thus knew how to reproduce.

> My only thought about the downside of this is that an exiting task
> that gets stuck somewhere between setting PF_EXITING and calling
> cgroup_exit() won't show up in its cgroup's tasks file, since we'll
> enable cgroup links but skip it. I guess that's not a big deal.
>

Agree. I think it won't be a problem.

> Maybe it would be better to not do a cgroup_exit() until we're
> unhashed, so that cgroup_enable_task_cg_list() can't find the exiting
> task?
>
> Paul
>