Subject: Re: [PATCH][usercr]: Ghost tasks must be detached
Posted by Oren Laadan on Wed, 09 Feb 2011 12:18:49 GMT
View Forum Message <> Reply to Message

On 02/09/2011 07:01 AM, Louis Rilling wrote:
> On 08/02/11 18:09 -0800, Sukadev Bhattiprolu wrote:
>> Oren Laadan [orenl@cs.columbia.edu] wrote:
>> |
>> |
>> | On 02/05/2011 04:40 PM, Sukadev Bhattiprolu wrote:
>> | > Oren Laadan [orenl@cs.columbia.edu] wrote:
>> | > | Suka,
>> | > |
>> | > | This patch - and the corresponding kernel patch - are wrong
>> | >
>> | > Ah, I see that now.
>> | >
>> | > But am not sure about the kernel part though. We were getting a crash
>> | > reliably (with older kernels) because of the ->exit_signal = -1 in
>> | > do_ghost_task().
>> |
>> | Are we still getting it with 2.6.37 ?
>>
>> I am not currently getting the crash on 2.6.37 - I thought it was due to
>> the following commit which removed the check for task_detached() in
>> do_wait_thread().
>>
>> commit 9cd80bbb07fcd6d4d037fad4297496d3b132ac6b
>> Author: Oleg Nesterov <oleg@redhat.com>
>> Date: Thu Dec 17 15:27:15 2009 -0800
>
> I don't think that this introduced the bug. The bug triggers with EXIT_DEAD
> tasks, for which wait() must ignore (see below). So, the bug looks still there
> in 2.6.37.
>
>>
>> But if that is true, I need to investigate why Louis Rilling was getting
>> the crash in Jun 2010 - which he tried to fix here:
>>
>> http://lkml.org/lkml/2010/6/16/295
>
> I was getting the crash on Kerrighed, which heavily patches the 2.6.30 kernel.
> I could reproduce it on vanilla Linux of the moment (2.6.35-rc3), but
> only after introducing artificial delays in release_task().
>
> IIRC, what triggers the crash is some exiting detached task in the
> pid_namespace, which goes EXIT_DEAD, and as such cannot be reaped by
> zap_pid_ns_processes()->sys_wait4(). So with some odd timing, the detached

```
> task can call proc_flush_task() after container init does, which triggers the
> proc_mnt crash.
>
> Container init                    Some detached task in the ctnr
>                                 exit_notify()
>      ->exit_state = EXIT_DEAD
> exit_notify()
>  forget_original_parent()
>   find_new_reaper()
>    zap_pid_ns_processes()
>     sys_wait4()
>      /* cannot reap EXIT_DEAD tasks */
>  /* reparents EXIT_DEAD tasks to global init */
>
> Container reaper
> release_task()
>  proc_flush_task()
>   pid_ns_release_proc()
>                                 release_task()
>                                  proc_flush_task()
>                                   proc_flush_task_mnt()
>                                    KABOOM
```

Louis, thanks for the explanation, and two follow-up questions:

1) Is there a patch circulating for this ? or even better, on the way to mainline ?

2) Would it suffice if the c/r code ensures that the init never exits before any EXIT_DEAD tasks ?

Thanks,

Oren.