Subject: Re: [PATCH][usercr]: Ghost tasks must be detached
Posted by Oren Laadan on Wed, 09 Feb 2011 03:35:20 GMT
View Forum Message <> Reply to Message

On 02/08/2011 09:09 PM, 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
>
> 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 see. So basically there is a kerenl bug that can be potentially
exposed by the c/r code. Therefore, we need to fix the kernel bug...
(and until such a fix makes it to mainline, we'll add it as part of
the linux-cr patchset).

>
> Even if we are not currently not getting the crash, I think user-space
> actions can result in the container-init being unable to forcibly kill
> all its children and exit.
>
> Eg: if ghost tasks are pushed into a child pid namespace (by intentionally
> setting ->piddepth in usercr/restart.c), we can have a situation where the
> ghost task exits silently, the parent (i.e container-init can be left hanging).

I don't quite understand what you mean here. Basically, the ghost tasks
are only alive _during_ restart and are gone when the restart completes.
Therefore they cannot affect whether the init task of the new pidns will
hang or terminate -- that init task has no knowledge of ghost tasks.

Let's consider the two possible scenarios:
(1) container restart (that includes the container init)
(2) subtree restart in a new pidns (that does not include the init task,
and instead user-cr provides an init process to hold the new pidns alive).

Case 1: the (restarted) init task was part of the checkpoint. Typically
it would "wait()" in a loop for children until it gets ECHILD and then
exits (the container). Ghost tasks are not a factor here.

Case 2: the (injected) init task was not part of the checkpoint. It does
the same as the typical init in case 1: loop until wait() says no more
children, then exits. In this case, there will be at least one child of
that init task, because at least one task was restarted ... Typically,
when that child exits, our injected init task will exit. Again, ghost
tasks do no participate.

>
> It can be argued that the incorrect changes in usercr code result in the
> application hang.
>
> But pid namespace is supposed to guarantee that if a container-init is
> terminated, it will take the pid namespace down. But some userspace
> actions can result in kill -9 of container-init leaving the container-init
> hung forever.

So I guess I don't quite understand the concern. Can you describe a
concrete example ?

>
> | >
> | > One fix I was watching for was Eric Biederman's
> | >
> | > http://lkml.org/lkml/2010/7/12/213
> | >
> | > which AFAICT has not been merged yet.
> |
> | If we need it and it isn't in mainline (any reason why ?) then
> | we can just add it to our linux-cr tree, as a preparatory patch.
> |
> | >
> | > Was there another change to 2.6.37 that would prevent the crash ?
> |

> | I don't know whether *that* crash still happens in 2.6.37 -
> | because I still didn't test it with that kernel line back.
> | (Actually, I never experienced that crash here even with
> | earlier kernels).
>
> Yes, it needed some "accidental" usercr change to expose the crash :-)
>
> (I will try to send a patch to existing usercr and a test case to repro
> this problem)
>

Thanks,

Oren.