
Subject: Re: [PATCH 1/15] Move exit_task_namespaces()
Posted by [Pavel Emelianov](#) on Mon, 06 Aug 2007 11:21:46 GMT
[View Forum Message](#) <> [Reply to Message](#)

Oleg Nesterov wrote:

> On 08/06, Pavel Emelianov wrote:

>> Oleg Nesterov wrote:

>>> On 08/06, Pavel Emelianov wrote:

>>>> Oleg Nesterov wrote:

>>>>> On 07/26, Pavel Emelianov wrote:

>>>>>> The reason to release namespaces after reparenting is that when task

>>>>>> exits it may send a signal to its parent (SIGCHLD), but if the parent

>>>>>> has already exited its namespaces there will be no way to decide what

>>>>>> pid to defer to him - parent can be from different namespace.

>>>>> I almost forgot about this one...

>>>>>

>>>>> After reading the whole series, I can't understand the above explanation

>>>>> any longer. The parent can't be from different namespace, either we have

>>>>> another sub-thread, or we reparent the child to /sbin/init which should

>>>>> be from the same namespace.

>>>> If the child that is a new namespace's init is exiting its parent is from

>>>> the

>>>> different namespace.

>>> In that case it doesn't have childs. They were SIGKILL'ed before

>>> exit_notify().

>> It does not, but its parent - does :)

>

> Yes. But in that case forget_original_parent() is no-op! This means that

> it is not needed to move exit_task_namespace() after.

>

>>>> Moreover, we will probably want to implement "entering"

>>>> the pid namespace, so having tasks with parents from another namespace

>>>> will

>>>> be OK.

>>> Well. I saw this word "entering", but I don't know the meaning. Just

>>> curious,

>>> could you explain?

>> "Entering" means "moving task to arbitrary namespace"

>

> Heh. Very much nontrivial, good luck :) At least we need to grow ->numbers[],

> and if its pid was pinned...

>

>>> And, if an exiting task has a child which is already from another

>>> namespace,

>>> why can't we release our namespace before re-parenting? I guess I need to

>>> know what "entering" means to understand this...

>> One of the desired actions was the following:

>> 1. task X clones the new namespace with the child Y as this namespace's

>> init;
>> 2. task X waits for SIGCHLD to come informing that the namespace is dead.
>> In this scenario we need to set the Y's pid as it is seen from X's
>> namespace in siginfo.
>
> Yes sure. But again, how this is connected to "we should do exit_task_namespace()
> after forget_original_parent()" ?
>
> I guess I missed something stupid and simple...

If task X is exiting and has already exit_task_namespaces()-ed task
Y will OOPs during its exit in determining parent's namespace. I agree
that in that case this is not important what namespace X belongs to,
but we need to handle the race with changing the nsproxy from not-NULL
to NULL. This is OK to make this under task_lock() but what to add
extra locking for if we can avoid it?

> Oleg.
>
>
