Subject: Re: [PATCH 2/6] IPC namespace - utils
Posted by Cedric Le Goater on Tue, 13 Jun 2006 21:17:44 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:

>> task records a list of struct sem_undo each containing a semaphore id. When
>> we unshare ipc namespace, we break the 'reference' between the semaphore id
>> and the struct sem_array because the struct sem_array are cleared and freed
>> in the new namespace. When the task exit, that inconstency could lead to
>> unexpected results in exit_sem(), task locks, BUG_ON, etc. Nope ?
>
> Agreed. Hmm. I bet I didn't see this one earlier because it is specific
> to the unshare case. In this case I guess we should either deny the unshare
> or simply undo all of the semaphores. Because we will never be able to
> talk to them again.

So aren't we reaching the unshare() limits ? Shouldn't we be using the
exec() principle for the sysvipc namespace ? clear it all and start from
scratch.

> Thinking about this some more we need to unsharing the semaphore undo semantics
> when we create a new instances of the sysvipc namespace. Which means that
> until that piece is implemented we can't unshare the sysvipc namespace.

no big issue I think. exit_sem() does it already. it would end up coding
the yet unsupported unshare_semundo().

> But we clearly need the check in check_unshare_flags and the start of copy_process.

Yes. CLONE_SYSVSEM and CLONE_NEWIPC overlap in some ways.

>>>> * I don't like the idea of being able to unshare the ipc namespace and keep
>>>> some shared memory from the previous ipc namespace mapped in the process mm.
>>>> Should we forbid the unshare ?
>>> No. As long as the code handles that case properly we should be fine.
>> what is the proper way to handle that case ? the current patchset is not
>> protected : a process can be in one ipc namespace and use a shared segment
>> from a previous ipc namespace. This situation is not desirable in a
>> migration scenario. May be asking too much for the moment ... and I agree
>> this can be fixed by the way namespaces are created.
>
> As long as the appropriate reference counting happens it shouldn't be
> a problem. We obviously can't use the sysvipc name of the shm area
> but mmap and reads and writes should continue to work.

in that case, namespace ids are protected but namespace objects aren't. I
expect a higher level object (container) making sure this is consistent.

C.