Subject: Re: [PATCH 2/6] IPC namespace - utils
Posted by ebiederm on Mon, 12 Jun 2006 21:49:03 GMT
View Forum Message <> Reply to Message

Cedric Le Goater <clg@fr.ibm.com> writes:

> Eric W. Biederman wrote:
>> Cedric Le Goater <clg@fr.ibm.com> writes:
>>
>>> I've used the ipc namespace patchset in rc6-mm2. Thanks for putting this
>>> together, it works pretty well ! A few questions when we clone :
>>>
>>> * We should do something close to what exit_sem() already does to clear the
>>> sem_undo list from the task doing the clone() or unshare().
>>
>> Possibly which case are you trying to prevent?
>
> 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.

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.

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

>>> * 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.

>> As a general principle we should be able to keep things from other namespaces
>> open if we get them. The chroot or equivalent binary is the one that needs
>> to ensure these kinds of issues don't exist if we care.
>>
>> Speaking of we should put together a small test application probably similar
>> to chroot so people can access these features at least for testing.
>
> are you thinking about a command unshare()ing each namespace or some kind
> of create_nsproxy ?

A little user space program like chroot. That takes a flag of which
namespaces not to share. I have one around somewhere. Just enough
of something that these interfaces can be exercised from userspace.

Eric