
Subject: Re: [PATCH 2.6.24-rc8-mm1 09/15] (RFC) IPC: new kernel API to change an ID

Posted by [dev](#) on Thu, 31 Jan 2008 09:54:10 GMT

[View Forum Message](#) <> [Reply to Message](#)

Why user space can need this API? for checkpointing only?

Then I would not consider it for inclusion until it is clear how to implement checkpointing.

As for me personally - I'm against exporting such APIs, since they are not needed in real-life user space applications and maintaining it forever for compatibility doesn't worth it.

Also such APIs allow creation of non-GPL checkpointing in user-space, which can be of concern as well.

Kirill

Pierre Peiffer wrote:

> Hi again,

>

> Thinking more about this, I think I must clarify why I choose this way.

> In fact, the idea of these patches is to provide the missing user APIs (or

> extend the existing ones) that allow to set or update `_all_` properties of all

> IPCs, as needed in the case of the checkpoint/restart of an application (the

> current user API does not allow to specify an ID for a created IPC, for

> example). And this, without changing the existing API of course.

>

> And `msgget()`, `semget()` and `shmget()` does not have any parameter we can use to

> specify an ID.

> That's why I've decided to not change these routines and add a new control

> command, `IPC_SETID`, with which we can change the ID of an IPC. (that looks to

> me more straightforward and logical)

>

> Now, this patch is, in fact, only a preparation for the patch 10/15 which

> really complete the user API by adding this `IPC_SETID` command.

>

> (... continuing below ...)

>

> Alexey Dobriyan wrote:

>> On Tue, Jan 29, 2008 at 05:02:38PM +0100, pierre.peiffer@bull.net wrote:

>>> This patch provides three new API to change the ID of an existing

>>> System V IPCs.

>>>

>>> These APIs are:

>>> `long msg_chid(struct ipc_namespace *ns, int id, int newid);`

>>> `long sem_chid(struct ipc_namespace *ns, int id, int newid);`

>>> `long shm_chid(struct ipc_namespace *ns, int id, int newid);`

>>>

>>> They return 0 or an error code in case of failure.

```

>>>
>>> They may be useful for setting a specific ID for an IPC when preparing
>>> a restart operation.
>>>
>>> To be successful, the following rules must be respected:
>>> - the IPC exists (of course...)
>>> - the new ID must satisfy the ID computation rule.
>>> - the entry in the idr corresponding to the new ID must be free.
>>> ipc/util.c      | 48 ++++++
>>> ipc/util.h      | 1 +
>>> 8 files changed, 197 insertions(+)
>> For the record, OpenVZ uses "create with predefined ID" method which
>> leads to less code. For example, change at the end is all we want from
>> ipc/util.c .
>
> And in fact, you do that from kernel space, you don't have the constraint to fit
> the existing user API.
> Again, this patch, even if it presents a new kernel API, is in fact a
> preparation for the next patch which introduces a new user API.
>
> Do you think that this could fit your need ?
>

```

Containers mailing list
Containers@lists.linux-foundation.org
<https://lists.linux-foundation.org/mailman/listinfo/containers>
