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 13:11:26 GMT
View Forum Message <> Reply to Message

Pierre,

my point is that after you've added interface "set IPCID", you'll need more and more for checkpointing:
- "create/setup conntrack" (otherwise connections get dropped),
- "set task start time" (needed for Oracle checkpointing BTW),
- "set some statistics counters (e.g. networking or taskstats)"
- "restore inotify"
and so on and so forth.

Exporting such intimate kernel interfaces to user space doesn't look sane.
Exactly from compatibility and maintenance POV. You'll be burden with supporting them for a long time.
Remember recent story with SLUB and /proc/slabinfo?

Hope I made my argument more clear this time.

Thanks,
Kirill

Pierre Peiffer wrote:
>
> Kirill Korotaev wrote:
>> Why user space can need this API? for checkpointing only?
>
> I would say "at least for checkpointing"... ;) May be someone else may find an
> interest about this for something else.
> In fact, I'm sure that you have some interest in checkpointing; and thus, you
> have probably some ideas in mind; but whatever the solution you will propose,
> I'm pretty sure that I could say the same thing for your solution.
> And what I finally think is: even if it's for "checkpointing only", if many
> people are interested by this, it may be sufficient to push this ?
>
>> 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.
>
> Maintaining these patches is not a big deal, really, but this is not the main
> point; the "need in real life" (1) is in fact the main one, and then, the "is
> this solution the best one ?" (2) the second one.
>
> About (1), as said in my first mail, as the namespaces and containers are being

> integrated into the mainline kernel, checkpoint/restart is (or will be) the next
> need.
> About (2), my solution propose to do that, as much as possible from userspace,
> to minimize the kernel impact. Of course, this is subject to discussion. My
> opinion is that doing a full checkpoint/restart from kernel space will need lot
> of new specific and intrusive code; I'm not sure that this will be acceptable by
> the community. But this is my opinion only. Discusion is opened.
>
>> Also such APIs allow creation of non-GPL checkpointing in user-space, which can be of
concern as well.
>
> Honestly, I don't think this really a concern at all. I mean: I've never seen
> "this allows non-GPL binary and thus, this is bad" as an argument to reject a
> functionality, but I may be wrong, and thus, it can be discussed as well.
> I think the points (1) and (2) as stated above are the key ones.
>
> Pierre
>
>> 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, IP_SETID, with which we can 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