
Subject: Re: [RFC][PATCH 6/7] Account for the number of tasks within container
Posted by [Paul Menage](#) on Thu, 08 Mar 2007 13:49:27 GMT
[View Forum Message](#) <> [Reply to Message](#)

On 3/6/07, Pavel Emelianov <xemul@sw.ru> wrote:

> The idea is:
>
> Task may be "the entity that allocates the resources" and "the
> entity that is a resource allocated".
>
> When task is the first entity it may move across containers
> (that is implemented in your patches). When task is a resource
> it shouldn't move across containers like files or pages do.
>
> More generally - allocated resources hold reference to original
> container till they die. No resource migration is performed.
>
> Did I express my idea cleanly?

Yes, but I disagree with the premise. The title of your patch is "Account for the number of tasks within container", but that's not what the subsystem does, it accounts for the number of forks within the container that aren't directly accompanied by an exit.

Ideally, resources like files and pages would be able to follow tasks as well. The reason that files and pages aren't easily migrated from one container to another is that there could be sharing involved; figuring out the sharing can be expensive, and it's not clear what to do if two users are in different containers.

But in the case of a task count, there are no such issues with sharing, so it seems to me to be more sensible (and more efficient) to just limit the number of tasks in a container.

i.e. when moving a task into a container or forking a task within a container, increment the count; when moving a task out of a container or when it exits, decrement the count.

With your approach, if you were to set the task limit of an empty container A to 1, and then move a process P from B into A, P would be able to fork a new child, since the "task count" would be 0 (as P was being charged to B still). Surely the fact that there's 1 process in A should prevent P from forking?

Paul
