

---

Subject: Re: [PATCH 03/33] task containersv11 add tasks file interface

Posted by [Paul Menage](#) on Wed, 03 Oct 2007 15:16:53 GMT

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

---

On 10/3/07, Paul Jackson <pj@sgi.com> wrote:

>  
> - What are these apparent 'exec notifications' that are provided to  
> user space that the following mentions - I cannot find any other  
> mention of them:  
>  
> With the ability to classify tasks differently for different  
> resources (by putting those resource subsystems in different  
> hierarchies) then the admin can easily set up a script which  
> receives exec notifications and depending on who is launching  
> the browser he can

It's the process connector netlink notifier. It can report  
fork/exit/exec/setuid events to userspace. See  
drivers/connector/cn\_proc.c

>  
>  
> - It states in cgroups.txt:  
>  
> \*\*\* notify\_on\_release is disabled in the current patch set. It will be  
> \*\*\* reactivated in a future patch in a less-intrusive manner  
>  
> This doesn't seem to be true, and had better not be true.  
> From what I can tell, notify\_on\_release still works for cpusets,  
> and it is important that it continue to work when cgroups are  
> folded into the main line kernel.

Correct, it's reactivated in a later patch in the series, but this  
intermediate comment snuck through.

>  
> Each cgroup object created by the system has an array of pointers,  
> indexed by subsystem id; this pointer is entirely managed by the  
> subsystem; the generic cgroup code will never touch this pointer.  
>  
> Is plural "pointers", or singular "pointer", the correct wording?

Probably plural.

>  
> - Several lines near the end of cgroups.txt start with "LL".  
> I guess they list what locks are held while taking the call,  
> but the notation seems cryptic and unfamiliar to me, and its

> meaning here undocumented.

"Locking Level", describing which locks *\*are\** held, and which are *\*not\** held during a call. I thought it was a more generally widely-used commenting convention, but I don't see any other uses of it in the kernel. I can replace them with "holds cgroup\_mutex" or "doesn't hold cgroup\_mutex" for clarity.

>  
> - There are many instances of the local variable 'cont', referring  
> to a struct cgroup pointer. I presume the spelling 'cont' is a  
> holdover from the time when we called these containers.

Yes, and since cgroup is short for "control group", "cont" still seemed like a reasonable abbreviation. (And made the automatic renaming much simpler).

>  
> - The code in attach\_task which skips the attachment of a task to  
> the group it is already in has to be removed. Cpusets depends  
> on reattaching a task to its current cpuset, in order to trigger  
> updating the cpus\_allowed mask in the task struct. This is a  
> hack, granted, but an important one. It avoids checking for a  
> changed cpuset 'cpus' setting in critical scheduler code paths.

I don't quite understand how this is meant to work - under what circumstances would it occur? Are there cases when userspace is required to try to reattach a task to its current cpuset in order to get a cpu mask change to stick?

Other comments noted, thanks.

Paul

---

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

---