Subject: Re: [RFC][PATCH 6/7] Account for the number of tasks within container Posted by xemul on Wed, 07 Mar 2007 07:10:56 GMT

View Forum Message <> Reply to Message

Paul Menage wrote:

> Hi Pavel, > > On 3/6/07, Pavel Emelianov <xemul@sw.ru> wrote: >> diff -upr linux-2.6.20.orig/include/linux/sched.h >> linux-2.6.20-0/include/linux/sched.h >> --- linux-2.6.20.orig/include/linux/sched.h 2007-03-06 >> 13:33:28.00000000 +0300 2007-03-06 >> +++ linux-2.6.20-0/include/linux/sched.h >> 13:33:28.00000000 +0300 >> @ @ -1052,6 +1055,9 @ @ struct task_struct { >> #ifdef CONFIG_FAULT_INJECTION int make it fail; >> >> #endif >> +#ifdef CONFIG PROCESS CONTAINER struct numproc_container *numproc_cnt; >> + >> +#endif >> }; > > Why do you need a pointer added to task_struct? One of the main points > of the generic containers is to avoid every different subsystem and > resource controller having to add new pointers there. > >> + >> + rcu read lock(); np = numproc_from_cont(task_container(current, &numproc_subsys)); >> + css get current(&np->css); >> + > > There's no need to hold a reference here - by definition, the task's > container can't go away while the task is in it. > > Also, shouldn't you have an attach() method to move the count from one > container to another when a task moves? 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?

> Paul

>

Page 2 of 2 ---- Generated from OpenVZ Forum