Subject: Re: [ckrm-tech] [patch00/05]: Containers(V2)- Introduction
Posted by Paul Menage on Wed, 20 Sep 2006 17:23:54 GMT
View Forum Message <> Reply to Message

On 9/20/06, Nick Piggin <nickpiggin@yahoo.com.au> wrote:

> Yeah, I'm not sure about that. I don't think really complex schemes
> are needed... but again I might need more knowledge of their workloads
> and problems.
>

The basic need for separating files into containers distinct from the tasks that are using them arises when you have several "jobs" all working with the same large data set. (Possibly read-only data files, or possibly one job is updating a dataset that's being used by other jobs).
For automated job-tracking and scheduling, it's important to be able to distinguish shared usage from individual usage (e.g. to be able to answer questions "if I kill job X, how much memory do I get back?" and "how do I recover 1G of memory on this machine")

As an example, assume two jobs each with 100M of anonymous memory both mapping the same 1G file, for a total usage of 1.2G.

Any setup that doesn't let you distinguish shared and private usage makes it hard to answer that kind of scheduling questions. E.g.:

- first user gets charged for the page -> first job reported as 1.1G, and the second as 0.1G.

- page charges get shared between all users of the page -> two tasks using 0.6G each.

- all users get charged for the page -> two tasks using 1.1G each.

But in fact killing either one of these jobs individually would only free up 100M

By explicitly letting userspace see that there are two jobs each with a private usage of 100M, and they're sharing a dataset of 1G, it's possible to make more informed decisions.

The issue of telling the kernel exactly which files/directories need to be accounted separately can be handled by userspace.

It could be done by per-page accounting, or by constraining particular files to particular memory zones, or by just tracking/limiting the number of pages from each address_space in the pagecache, but I think

that it's important that the kernel at least provide the primitive support for this.

Paul