Subject: Re: Re: [ckrm-tech] [PATCH] BC: resource beancounters (v4) (added user memory)

Posted by kir on Mon, 11 Sep 2006 19:47:24 GMT

View Forum Message <> Reply to Message

Rohit Seth wrote:
> On Mon, 2006-09-11 at 11:25 -0700, Chandra Seetharaman wrote:
>
>> On Fri, 2006-09-08 at 14:43 -0700, Rohit Seth wrote:
>> <snip>
>>
>>
>>>>> Guarantee may be one of
>>>>>
>>>>>   1. container will be able to touch that number of pages
>>>>>   2. container will be able to sys_mmap() that number of pages
>>>>>   3. container will not be killed unless it touches that number of pages
>>>>>   4. anything else
>>>>>
>>>> I would say (1) with slight modification
>>>>   "container will be able to touch _at least_ that number of pages"
>>>>
>>>>
>>> Does this scheme support running of tasks outside of containers on the
>>> same platform where you have tasks running inside containers. If so
>>> then how will you ensure processes running out side any container will
>>> not leave less than the total guaranteed memory to different containers.
>>>
>>>
>> There could be a default container which doesn't have any guarantee or
>> limit.
>>
>
> First, I think it is critical that we allow processes to run outside of
> any container (unless we know for sure that the penalty of running a
> process inside a container is very very minimal).
>
(1) there is a set of processes running outside of any container. In
OpenVZ we call that "VE0" or "host system", probably Chandra meant that
by "default container".
(2) The host system is used to manage the containers (start/stop/set
parameters/create/destroy).
(3) the penalty of running a process inside a container is indeed very low.

> And anything running outside a container should be limited by default
> Linux settings.
>
(4) due to (2), it is not recommended to run anything but the tasks used

to manage the containers -- otherwise your gonna have security problems
(5) "Default Linux settings" do not cover everything (for example --
dentry cache), thus the need for beancounters.
>> When you create containers and assign guarantees to each of them
>> make sure that you leave some amount of resource unassigned.
>>
>   ^^^^^ This will force the "default" container
> with limits (indirectly). IMO, the whole guarantee feature gets defeated
> the moment you bring in this fuzziness.
>
>
>> That
>> unassigned resources can be used by the default container or can be used
>> by containers that want more than their guarantee (and less than their
>> limit). This is how CKRM/RG handles this issue.
>>
>>
>>
>
> It seems that a single notion of limit should suffice, and that limit
> should more be treated as something beyond which that resource
> consumption in the container will be throttled/not_allowed.
>
Beancounters have a notion of "barrier" and "limit". For some parameters
they are the same, but for some parameters they differ -- and there is
some "safety gap" between the barrier and the limit. The problem is for
some types of resources you can not throttle or deny -- the only way is
to kill the process. The one (but not the only one) example is process
stack expansion. See more at http://wiki.openvz.org/UBC (and follow the
menu at the right side).