Subject: Re: [ckrm-tech] [PATCH] BC: resource beancounters (v4) (added user memory)
Posted by Pavel Emelianov on Mon, 18 Sep 2006 11:32:32 GMT
View Forum Message <> Reply to Message

Balbir Singh wrote:
> Pavel Emelianov wrote:
>> Balbir Singh wrote:
>>
>> [snip]
>>
>>> This approach has the following disadvantages
>>> 1. Lets consider initialization - When we create 'n' groups
>>> initially, we need
>>>   to spend O(n^2) time to assign guarantees.
>>
>> 1. Not guarantees - limits. If you do not need guarantees - assign
>>   overcommited limits. Most of OpenVZ users do so and nobody claims.
>> 2. If you start n groups at once then limits are calculated in O(n)
>>   time, not O(n^2).
>
> Yes.. if you start them at once, but if they are incrementally
> added and started it is O(n^2)

See my comment below.

>
>>
>>> 2. Every time a limit or a guarantee changes, we need to recalculate
>>> guarantees
>>>   and ensure that the change will not break any guarantees
>>
>> The same.
>>
>>> 3. The same thing as stated above, when a resource group is created
>>> or deleted
>>>
>>> This can lead to some instability; a change in one group propagates to
>>> all other groups.
>>
>> Let me cite a part of your answer on my letter from 11.09.2006:
>> "...
>>   xemul> I have a node with 1Gb of ram and 10 containers with 100Mb
>>   xemul> guarantee each. I want to start one more.
>>   xemul> What shall I do not to break guarantees?
>>
>> Don't start the new container or change the guarantees of the
>> existing ones to accommodate this one ... It would be perfectly

>> ok to have a container that does not care about guarantees to
>> set their guarantee to 0 and set their limit to the desired value
>> ..."
>>
>> The same for the limiting - either do not start new container, or
>> recalculate limits to meet new requirements. You may not take care of
>> guarantees as weel and create an overcommited configuration.

As I do not see any reply on this I consider "O(n^2) disadvantage" to
be irrelevant.

>>
>> And one more thing. We've asked it many times and I ask it again -
>> please, show us the other way for providing guarantee rather than
>> limiting or reserving.
>
> There are some other options, I am sure Chandra will probably have
> more.
>
> 1. Reclaim resources from other containers. This can be done well for
>   user-pages, if we ensure that each container does not mlock more
>   than its guaranteed share of memory.

We've already agreed to consider unreclaimable resources only.
If we provide reclaimable memory *only* then we can provide any
guarantee with a single page available for user-space.
Unreclaimable resource is the most interesting one.

> 2. Provide best effort guarantees for non-reclaimable memory

That's the question - how?

> 3. oom-kill a container or a task within a resource group that has
>   exceeded its guarantee and some other container is unable to meet its
>   guarantee

Oom-killer must start only when there are no other ways to find memory.
This must be a "last argument", not the regular solution.