Subject: Re: [ckrm-tech] [RFC][PATCH] UBC: user resource beancounters
Posted by Chandra Seetharaman on Fri, 18 Aug 2006 19:39:52 GMT
View Forum Message <> Reply to Message

Kirill,

Here are some concerns I have (as of now) w.r.t using UBC for resource management (in the context of resource groups).

- guarantee support is missing. I do not see any code to provide the minimum amount of resource a group can get. It is important for providing QoS. (In a different email you did mention guarantee, i am referring it here for completeness).
- Creation of a UBC and assignment of task to a UBC always happen in the context of the task that is affected. I can understand it works in OpenVZ environment, but IMO has issues if one wants it to be used for basic resource management
  - application needs to be changed to use this feature.
  - System administrator does not have the control to assign tasks to a UBC. Application does by itself.
  - Assignment of task to a UBC need to be transparent to the application.
- UBC is deleted when the last task (in that UBC) exits. For resource management purposes, UBC should be deleted only when the administrator deletes it.
- No ability to set resource specific configuration information.
- No ability to maintain resource specific data in the controller.
- No ability to get the list of tasks belonging to a UBC.
- Doesn't inform the resource controllers when limits(shares) change.
- Doesn't inform the resource controllers when a task's UBC has changed.
- Doesn't recalculate the resource usage when a task's UBC has changed. i.e doesn't uncharge the old UBC and charge new UBC.
- For a system administrator name for identification of a UBC is better than a number (uid).

regards,

chandra

On Wed, 2006-08-16 at 19:24 +0400, Kirill Korotaev wrote:
> The following patch set presents base of
> User Resource Beancounters (UBC).
> UBC allows to account and control consumption
> of kernel resources used by group of processes.
>
> The full UBC patch set allows to control:
> - kernel memory. All the kernel objects allocatable
> on user demand should be accounted and limited

> for DoS protection.
> E.g. page tables, task structs, vmas etc.
>
> - virtual memory pages. UBC allows to
> limit a container to some amount of memory and
> introduces 2-level OOM killer taking into account
> container's consumption.
> pages shared between containers are correctly
> charged as fractions (tunable).
>
> - network buffers. These includes TCP/IP rcv/snd
> buffers, dgram snd buffers, unix, netlinks and
> other buffers.
>
> - minor resources accounted/limited by number:
> tasks, files, flocks, ptys, siginfo, pinned dcache
> mem, sockets, iptentries (for containers with
> virtualized networking)
>
> As the first step we want to propose for discussion
> the most complicated parts of resource management:
> kernel memory and virtual memory.
> The patch set to be sent provides core for UBC and
> management of kernel memory only. Virtual memory
> management will be sent in a couple of days.
>
> The patches in these series are:
> diff-ubc-kconfig.patch:
> Adds kernel/ub/Kconfig file with UBC options and
> includes it into arch Kconfigs
>
> diff-ubc-core.patch:
> Contains core functionality and interfaces of UBC:
> find/create beancounter, initialization,
> charge/uncharge of resource, core objects' declarations.
>
> diff-ubc-task.patch:
> Contains code responsible for setting UB on task,
> it's inheriting and setting host context in interrupts.
>
> Task contains three beancounters:
> 1. exec_ub - current context. all resources are charged
> to this beancounter.
> 2. task_ub - beancounter to which task_struct is charged
> itself.
> 3. fork_sub - beancounter which is inherited by
> task's children on fork
>

> diff-ubc-syscalls.patch:
> Patch adds system calls for UB management:
> 1. sys_getluid - get current UB id
> 2. sys_setluid - changes exec_ and fork_ UBs on current
> 3. sys_setublimit - set limits for resources consumtions
>
> diff-ubc-kmem-core.patch:
> Introduces UB_KMEMSIZE resource which accounts kernel
> objects allocated by task's request.
>
> Objects are accounted via struct page and slab objects.
> For the latter ones each slab contains a set of pointers
> corresponding object is charged to.
>
> Allocation charge rules:
> 1. Pages - if allocation is performed with __GFP_UBC flag - page
> is charged to current's exec_ub.
> 2. Slabs - kmem_cache may be created with SLAB_UBC flag - in this
> case each allocation is charged. Caches used by kmalloc are
> created with SLAB_UBC | SLAB_UBC_NOCHARGE flags. In this case
> only __GFP_UBC allocations are charged.
>
> diff-ubc-kmem-charge.patch:
> Adds SLAB_UBC and __GFP_UBC flags in appropriate places
> to cause charging/limiting of specified resources.
>
> diff-ubc-proc.patch:
> Adds two proc entries user_beancounters and user_beancounters_sub
> allowing to see current state (usage/limits/fails for each UB).
> Implemented via seq files.
>
> Patch set is applicable to 2.6.18-rc4-mm1
>
> Thanks,
> Kirill
>
>
> -- ------------
> Using Tomcat but need to do more? Need to support web services, security?
> Get stuff done quickly with pre-integrated technology to make your job easier
> Download IBM WebSphere Application Server v.1.0.1 based on Apache Geronimo
> http://sel.as-us.falkag.net/sel?cmd=lnk&kid=120709&b id=263057&dat=121642
> _____
> ckrm-tech mailing list
> https://lists.sourceforge.net/lists/listinfo/ckrm-tech
--

 -- ----------

Chandra Seetharaman | Be careful what you choose....
- sekharan@us.ibm.com | .....you may get it.
