Subject: [PATCH] BC: resource beancounters (v4) (added user memory)
Posted by dev on Tue, 05 Sep 2006 14:59:30 GMT
View Forum Message <> Reply to Message

Core Resource Beancounters (BC) + kernel/user memory control.

BC allows to account and control consumption
of kernel resources used by group of processes.

Draft UBC description on OpenVZ wiki can be found at
http://wiki.openvz.org/UBC_parameters

The full BC 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. BCs allow 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 BC and
management of kernel memory only. Virtual memory
management will be sent in a couple of days.

The patches in these series are:
diff-atomic-dec-and-lock-irqsave.patch
 introduce atomic_dec_and_lock_irqsave()

diff-bc-kconfig.patch:
 Adds kernel/bc/Kconfig file with UBC options and
 includes it into arch Kconfigs

diff-bc-core.patch:
 Contains core functionality and interfaces of BC:
 find/create beancounter, initialization,
 charge/uncharge of resource, core objects' declarations.

diff-bc-task.patch:
 Contains code responsible for setting BC on task,
 it's inheriting and setting host context in interrupts.

 Task contains three beancounters:
 1. exec_bc - current context. all resources are charged
             to this beancounter.
 2. fork_bc - beancounter which is inherited by
             task's children on fork

diff-bc-syscalls.patch:
 Patch adds system calls for BC management:
 1. sys_get_bcid   - get current BC id
 2. sys_set_bcid   - changes exec_ and fork_ BCs on current
 3. sys_set_bclimit - set limits for resources consumtions
 4. sys_get_bcstat - returns limits/usages/fails for BC

diff-bc-kmem-core.patch:
 Introduces BC_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_BC flag - page
    is charged to current's exec_bc.
 2. Slabs - kmem_cache may be created with SLAB_BC flag - in this
    case each allocation is charged. Caches used by kmalloc are
    created with SLAB_BC | SLAB_BC_NOCHARGE flags. In this case
    only __GFP_BC allocations are charged.

diff-bc-kmem-charge.patch:
 Adds SLAB_BC and __GFP_BC flags in appropriate places
 to cause charging/limiting of specified resources.

diff-bc-vmlocked-core.patch:
Introduces new resource BC_LOCKEDPAGES for accounting
of mlock-ed user pages.

diff-bc-vmlocked-charge.patch:

Places calls to BC core over the kernel to charge locked memory.

diff-bc-privvm.patch:
This patch instroduces new resource - BC_PRIVVMPAGES.
Privvmpages acointing is described in details in
http://wiki.openvz.org/User_pages_accounting

diff-bc-vmrss-prep.patch:
This patch intruduces small preparations for vmrss accounting
to make reviewing simpler.

diff-bc-vmrss-core.patch:
This is the core of vmrss accounting.
Pages are accounted in fractions and it is described in details in
http://wiki.openvz.org/RSS_fractions_accounting

diff-bc-vmrss-charge.patch:
Calls to vmrss core code over the kernel to do accounting.

Summary of changes from v3 patch set:

* Added basic user pages accounting (lockedpages/privvmpages)
* spell in Kconfig
* Makefile reworked
* EXPORT_SYMBOL_GPL
* union w/o name in struct page
* bc_task_charge is void now
* adjust minheld/maxheld splitted

Summary of changes from v2 patch set:

* introduced atomic_dec_and_lock_irqsave()
* bc_adjust_held_minmax comment
* added __must_check for bc_*charge* funcs
* use hash_long() instead of own one
* bc/Kconfig is sourced from init/Kconfig now
* introduced bcid_t type with comment from Alan Cox
* check for barrier <= limit in sys_set_bclimit()
* removed (bc == NULL) checks
* replaced memcpy in beancounter_findcrate with assignment
* moved check 'if (mask & BC_ALLOC)' out of the lock
* removed unnecessary memset()

Summary of changes from v1 patch set:

* CONFIG_BEANCOUNTERS is 'n' by default
* fixed Kconfig includes in arches

* removed hierarchical beancounters to simplify first patchset
* removed unused 'private' pointer
* removed unused EXPORTS
* MAXVALUE redeclared as LONG_MAX
* beancounter_findcreate clarification
* renamed UBC -> BC, ub -> bc etc.
* moved BC inheritance into copy_process
* introduced reset_exec_bc() with proposed BUG_ON
* removed task_bc beancounter (not used yet, for numproc)
* fixed syscalls for sparc
* added sys_get_bcstat(): return info that was in /proc
* cond_syscall instead of #ifdefs

Many thanks to Oleg Nesterov, Alan Cox, Matt Helsley and others
for patch review and comments.

Patch set is applicable to 2.6.18-rc5-mm1

Thanks,
Kirill
