Subject: [PATCH] BC: resource beancounters (v5) (added userpages reclamation)
Posted by Kirill Korotaev on Thu, 05 Oct 2006 15:27:47 GMT
View Forum Message <> Reply to Message

MAJOR CHANGES:
- phys pages limit hit leads to user pages reclamation
- bc can be changed on arbitrary task

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.

* diff-bc-syscalls.patch:
Patch adds system calls for BC management:
1. sys_get_bcid - get current BC id
2. sys_set_bcid - changes BC on task
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-vmpages-core:
Accounting of private pages (mappings), done per-vma.
Charged on mmap().

* diff-bc-rsspages-core:

Phys pages accounting. Charging is done on page touch
(page fault) and limit hit leads to reclamation.
Reclamation is done based on beancounter list of pages.

* diff-bc-userpages-charge:
Hooks across the kernel for user pages accounting.

* bcctl.c:
Tool for BC management. Allows changing BC, setting BC limits,
viewing current usages.

Summary of changes from v4 patch set:
* changed set of resources - kmemsize, privvmpages, physpages
* added event hooks for resources (init, limit hit etc)
* added user pages reclamation (bc_try_to_free_pages)
* removed pages sharing accounting - charge to first user
* task now carries only one BC pointer, simplified
* make set_bcid syscall move arbitrary task into BC
* resources are not recharged when task moves
* each vm_area_struct carries a BC pointer

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-mm3

Thanks,
Kirill