Subject: [PATCH 00/10] Task Containers(V11): Introduction
Posted by Paul Menage on Fri, 20 Jul 2007 18:31:52 GMT
View Forum Message <> Reply to Message

This is an update to the task containers patchset.

Changes since V10 (May 30th) include:

- Based on 2.6.22-rc6-mm1 (minus existing container patches, see below)

- Rolled in various fix/tidy patches contributed by akpm and others

- Reorganisation of the mount/unmount code to use sget(); the new
approach is modelled on the NFS superblock code. This fixes some
potential lock inversions pointed out by lockdep.

- Fix various lockdep warnings

- Changed the create() subsystem callback to return a pointer to the
new state object rather than updating the subsystem pointer in the
container directly.

- Changed container_add_file() to automatically prefix the subsystem
name (and a period) on to all container files unless the filesystem
is mounted with the "noprefix" option (intended for use by the
legacy cpuset filesystem emulation).

- Added a release_agent= mount option to allow the release agent path
to be specified at mount time.

- css_put() is now completely non-blocking

- css_get()/css_put() avoid taking/dropping reference counts on the
root state since this can't be freed anyway; this saves some atomic ops

API changes (for subsystem writers):
1) return your new css object from create() callback
2) remove the subsystem name prefix from your cftype structures
3) pass your subsystem pointer as an additional new parameter to
container_add_file() and container_add_files()

Still TODO:

- finalize the naming

- add a hash-table based lookup for css_group objects.

- use seq_file properly in container tasks files to avoid having to

allocate a big array for all the container's task pointers.

- add virtualization support to allow delegation to virtual servers

- fix a lockdep false-positive - container_mutex nests inside
  inode->i_mutex, but there's a point in the mount code where we need to
  lock a newly-created (and hence guaranteed unlocked) directory from
  within container_mutex.

- more subsystems

Generic Process Containers

There have recently been various proposals floating around for
resource management/accounting and other task grouping subsystems in
the kernel, including ResGroups, User BeanCounters, NSProxy
containers, and others. These all need the basic abstraction of being
able to group together multiple processes in an aggregate, in order to
track/limit the resources permitted to those processes, or control
other behaviour of the processes, and all implement this grouping in
different ways.

This patchset provides a framework for tracking and grouping processes
into arbitrary "containers" and assigning arbitrary state to those
groupings, in order to control the behaviour of the container as an
aggregate.

The intention is that the various resource management and
virtualization/container efforts can also become task container
clients, with the result that:

- the userspace APIs are (somewhat) normalised

- it's easier to test e.g. the ResGroups CPU controller in
  conjunction with the BeanCounters memory controller, or use either of
them as the resource-control portion of a virtual server system.

- the additional kernel footprint of any of the competing resource
  management systems is substantially reduced, since it doesn't need
  to provide process grouping/containment, hence improving their
  chances of getting into the kernel

The patch set is structured as follows:

1) Basic container framework - filesystem and tracking structures

2) Support for the "tasks" control file

3) Hooks for fork() and exit()

4) Support for the container_clone() operation

5) Add /proc reporting interface

6) Share container subsystem pointer arrays between tasks with the
same assignments

7) Support for a userspace "release agent", similar to the cpusets
release agent functionality

8) Make cpusets a container subsystem

9) Simple CPU Accounting example subsystem

10) Simple container debugging subsystem

It applies to 2.6.22-rc6-mm1, *minus* the following patches (available
from http://www.kernel.org/pub/linux/kernel/people/akpm/mm/broken
-out-2007-06-27-03-28.tar.gz)

containersv10-basic-container-framework.patch
containersv10-basic-container-framework-fix.patch
containersv10-basic-container-framework-fix-2.patch
containersv10-basic-container-framework-fix-3.patch
containersv10-example-cpu-accounting-subsystem.patch
containersv10-example-cpu-accounting-subsystem-fix.patch
containersv10-add-tasks-file-interface.patch
containersv10-add-tasks-file-interface-fix.patch
containersv10-add-tasks-file-interface-fix-2.patch
containersv10-add-fork-exit-hooks.patch
containersv10-add-fork-exit-hooks-fix.patch
containersv10-add-container_clone-interface.patch
containersv10-add-container_clone-interface-fix.patch
containersv10-add-procfs-interface.patch
containersv10-add-procfs-interface-fix.patch
containersv10-make-cpusets-a-client-of-containers.patch
containersv10-make-cpusets-a-client-of-containers-whitespace .patch
containersv10-share-css_group-arrays-between-tasks-with-same -container-memberships.patch
containersv10-share-css_group-arrays-between-tasks-with-same
-container-memberships-fix.patch
containersv10-share-css_group-arrays-between-tasks-with-same
-container-memberships-cpuset-zero-malloc-fix-for-new-contai ners.patch
containersv10-simple-debug-info-subsystem.patch
containersv10-simple-debug-info-subsystem-fix.patch

containersv10-simple-debug-info-subsystem-fix-2.patch
containersv10-support-for-automatic-userspace-release-agents .patch
containersv10-support-for-automatic-userspace-release-agents -whitespace.patch
add-containerstats-v3.patch
add-containerstats-v3-fix.patch
update-getdelays-to-become-containerstats-aware.patch
containers-implement-subsys-post_clone.patch
containers-implement-namespace-tracking-subsystem-v3.patch

Signed-off-by: Paul Menage <menage@google.com>

--