Subject: [PATCH] Simplify /proc/cgroups
Posted by menage on Thu, 18 Oct 2007 02:56:36 GMT
View Forum Message <> Reply to Message

This patch simplifies /proc/cgroups by removing pointers and some
debugging information, and simply presenting a list of subsystems,
which hierarchy they are part of (if any) and the number of cgroups
created for that subsystem. Hierarchy id is determined by the bitmask
of subsystem ids attached to that hierarchy.

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

Several people have commented that /proc/cgroups is too confusing or
contains strange information. Here's an attempt to simplify it. New
typical output looks like:

#subsys_name hierarchy num_cgroups
cpuset 1 2
cpuacct 10 1
debug 0 1
ns 10 1
memory 0 1
cpu 0 1

Maybe there should be more tabs so that the columns line up better?
But then it'll be out of line if people create subsystems with longer
names ...

 kernel/cgroup.c | 25 ++++----------------------
 1 file changed, 4 insertions(+), 21 deletions(-)

Index: container-2.6.23-mm1/kernel/cgroup.c
===
--- container-2.6.23-mm1.orig/kernel/cgroup.c
+++ container-2.6.23-mm1/kernel/cgroup.c
@@ -2403,31 +2403,14 @@ static int proc_cgroupstats_show(struct
  int i;
  struct cgroupfs_root *root;

+ seq_puts(m, "#subsys_name\thierarchy\tnum_cgroups\n");
  mutex_lock(&cgroup_mutex);
- seq_puts(m, "Hierarchies:\n");
- for_each_root(root) {
- struct cgroup_subsys *ss;
- int first = 1;
- seq_printf(m, "%p: bits=%lx cgroups=%d (", root,

```
-      root->subsys_bits, root->number_of_cgroups);
-  for_each_subsys(root, ss) {
-   seq_printf(m, "%s%s", first ? "" : ", ", ss->name);
-   first = false;
-  }
-  seq_putc(m, ')');
-  if (root->sb) {
-   seq_printf(m, " s_active=%d",
-       atomic_read(&root->sb->s_active));
-  }
-  seq_putc(m, '\n');
- }
- seq_puts(m, "Subsystems:\n");
  for (i = 0; i < CGROUP_SUBSYS_COUNT; i++) {
   struct cgroup_subsys *ss = subsys[i];
-  seq_printf(m, "%d: name=%s hierarchy=%p\n",
-      i, ss->name, ss->root);
+  seq_printf(m, "%s\t%d\t%d\n",
+      ss->name, ss->root->subsys_bits,
+      ss->root->number_of_cgroups);
  }
- seq_printf(m, "Control Group groups: %d\n", css_set_count);
  mutex_unlock(&cgroup_mutex);
  return 0;
 }
```
