Subject: [PATCH -mm 2/3] cgroup: simplify init_subsys()
Posted by Li Zefan on Wed, 02 Apr 2008 02:16:52 GMT
View Forum Message <> Reply to Message

We are at system boot and there is only 1 cgroup group (i,e,
init_css_set), so we don't need to run through the css_set
linked list. Neither do we need to run through the task list,
since the init process hasn't been created yet.

Also referring to a comment in cgroup.h:

```
struct css_set
{
 ...
 /*
  * Set of subsystem states, one for each subsystem. This array
  * is immutable after creation apart from the init_css_set
  * during subsystem registration (at boot time).
  */
 struct cgroup_subsys_state *subsys[CGROUP_SUBSYS_COUNT];
}
```

Signed-off-by: Li Zefan <lizf@cn.fujitsu.com>
---
 kernel/cgroup.c | 30 ++++-------------------------
 1 files changed, 4 insertions(+), 26 deletions(-)

```
diff --git a/kernel/cgroup.c b/kernel/cgroup.c
index 78e5bde..2b72346 100644
--- a/kernel/cgroup.c
+++ b/kernel/cgroup.c
@@ -2467,7 +2467,6 @@ static int cgroup_rmdir(struct inode *unused_dir, struct dentry *dentry)
 static void __init cgroup_init_subsys(struct cgroup_subsys *ss)
 {
  struct cgroup_subsys_state *css;
- struct list_head *l;

  printk(KERN_INFO "Initializing cgroup subsys %s\n", ss->name);

@@ -2478,32 +2477,11 @@ static void __init cgroup_init_subsys(struct cgroup_subsys *ss)
  BUG_ON(IS_ERR(css));
  init_cgroup_css(css, ss, dummytop);

- /* Update all cgroup groups to contain a subsys
+ /* Update the init_css_set to contain a subsys
   * pointer to this state - since the subsystem is
-  * newly registered, all tasks and hence all cgroup
-  * groups are in the subsystem's top cgroup. */
```

```
- write_lock(&css_set_lock);
- l = &init_css_set.list;
- do {
-  struct css_set *cg =
-   list_entry(l, struct css_set, list);
-  cg->subsys[ss->subsys_id] = dummytop->subsys[ss->subsys_id];
-  l = l->next;
- } while (l != &init_css_set.list);
- write_unlock(&css_set_lock);
-
-  /* If this subsystem requested that it be notified with fork
-   * events, we should send it one now for every process in the
-   * system */
- if (ss->fork) {
-  struct task_struct *g, *p;
-
-  read_lock(&tasklist_lock);
-  do_each_thread(g, p) {
-   ss->fork(ss, p);
-  } while_each_thread(g, p);
-  read_unlock(&tasklist_lock);
- }
+  * newly registered, all tasks and hence the
+  * init_css_set is in the subsystem's top cgroup. */
+ init_css_set.subsys[ss->subsys_id] = dummytop->subsys[ss->subsys_id];

  need_forkexit_callback |= ss->fork || ss->exit;

--
1.5.4.rc3
```

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: [PATCH -mm 2/3] cgroup: simplify init_subsys()
Posted by Paul Menage on Wed, 02 Apr 2008 10:26:24 GMT
View Forum Message <> Reply to Message

On Tue, Apr 1, 2008 at 7:16 PM, Li Zefan <lizf@cn.fujitsu.com> wrote:

```
> -
> -       /* If this subsystem requested that it be notified with fork
> -        * events, we should send it one now for every process in the
> -        * system */
> -       if (ss->fork) {
> -               struct task_struct *g, *p;
> -
```

```
> -               read_lock(&tasklist_lock);
> -               do_each_thread(g, p) {
> -                       ss->fork(ss, p);
> -               } while_each_thread(g, p);
> -               read_unlock(&tasklist_lock);
> -       }
```

Should we maybe call ss->fork(ss, &init_task) ? Or just document that
ss->fork() explicitly doesn't get called for init.

Maybe also add some BUG()s to confirm that no other tasks have in fact
been forked by this point?

Paul

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: [PATCH -mm 2/3] cgroup: simplify init_subsys()
Posted by Li Zefan on Thu, 03 Apr 2008 04:59:34 GMT
View Forum Message <> Reply to Message

Paul Menage wrote:
> On Tue, Apr 1, 2008 at 7:16 PM, Li Zefan <lizf@cn.fujitsu.com> wrote:
```
>> -
>> -       /* If this subsystem requested that it be notified with fork
>> -        * events, we should send it one now for every process in the
>> -        * system */
>> -       if (ss->fork) {
>> -               struct task_struct *g, *p;
>> -
>> -               read_lock(&tasklist_lock);
>> -               do_each_thread(g, p) {
>> -                       ss->fork(ss, p);
>> -               } while_each_thread(g, p);
>> -               read_unlock(&tasklist_lock);
>> -       }
>
```
> Should we maybe call ss->fork(ss, &init_task) ? Or just document that
> ss->fork() explicitly doesn't get called for init.
>
> Maybe also add some BUG()s to confirm that no other tasks have in fact
> been forked by this point?
>

At system boot, before all the subsystems have been registered, no processes

including init has been forked, am I right? So the fork callback will be invoked when the init process is forked, so we don't need to call ss->fork(ss, &init_task).

I'll add a BUG_ON(), and update the document about the fork callback.

Regards,
Li Zefan

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers