Subject: Re: [PATCH] new cgroup controller "fork"
Posted by KAMEZAWA Hiroyuki on Thu, 17 Feb 2011 13:50:10 GMT
View Forum Message <> Reply to Message

On Thu, 17 Feb 2011 14:31:52 +0100
Max Kellermann <mk@cm4all.com> wrote:

> Can limit the number of fork()/clone() calls in a cgroup. It is
> useful as a safeguard against fork bombs.
>

brief comments below.

> Signed-off-by: Max Kellermann <mk@cm4all.com>
<snip>

```
> +static int
> +cgroup_fork_remaining_write(struct cgroup *cgroup, struct cftype *cft,
> +       s64 value)
> +{
> + struct cgroup_fork *t = cgroup_fork_group(cgroup);
> +
> + if (value < -1 || value > (1L << 30))
> +  return -EINVAL;
> +
> + spin_lock_irq(&t->lock);
> + t->remaining = (int)value;
> + spin_unlock_irq(&t->lock);
> +
> + return 0;
> +}
```

I wonder allowing to set the limit to Root cgroup may imply the system death.
How about disabling to set value to Root cgroup ?

```
> +
> +static const struct cftype cgroup_fork_files[] =  {
> + {
> +  .name = "remaining",
> +  .read_s64 = cgroup_fork_remaining_read,
> +  .write_s64 = cgroup_fork_remaining_write,
> + },
> +};
> +
> +static int
> +cgroup_fork_populate(struct cgroup_subsys *ss, struct cgroup *cgroup)
```

```
> +{
> + if (cgroup->parent == NULL)
> +  /* cannot limit the root cgroup */
> +  return 0;
> +
> + return cgroup_add_files(cgroup, ss, cgroup_fork_files,
> +    ARRAY_SIZE(cgroup_fork_files));
> +}
> +
> +struct cgroup_subsys fork_subsys = {
> + .name = "fork",
> + .create = cgroup_fork_create,
> + .destroy = cgroup_fork_destroy,
> + .fork = cgroup_fork_fork,
> + .populate = cgroup_fork_populate,
> + .subsys_id = fork_subsys_id,
> +};
> +
> +int
> +cgroup_fork_pre_fork(void)
> +{
> + struct cgroup_fork *t;
> + int err = 0;
> +
> + rcu_read_lock();
> +
> + t = cgroup_fork_current();
> + while (t->css.cgroup->parent != NULL && err == 0) {
> +  spin_lock_irq(&t->lock);
> +
> +  if (t->remaining == 0)
> +   err = -EPERM;
```

IIRC, fork()'s error code is EAGAIN or ENOMEM. The exisiting limit of rlimit() returns EAGAIN.

How about -EAGAIN here ? I think it's not good to add new error code for system calls.

Thanks,
-Kame

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

Subject: Re: [PATCH] new cgroup controller "fork"
Posted by Max Kellermann on Thu, 17 Feb 2011 14:09:25 GMT
View Forum Message <> Reply to Message

On 2011/02/17 14:50, KAMEZAWA Hiroyuki <kamezawa.hiroyu@jp.fujitsu.com> wrote:
> I wonder allowing to set the limit to Root cgroup may imply the system death.
> How about disabling to set value to Root cgroup ?

That is taken care of already:

> > +static int
> > +cgroup_fork_populate(struct cgroup_subsys *ss, struct cgroup *cgroup)
> > +{
> > + if (cgroup->parent == NULL)
> > + /* cannot limit the root cgroup */
> > + return 0;

The attribute simply doesn't exist in the root cgroup.

Also watch the loop condition in cgroup_fork_pre_fork() closely, the
root cgroup isn't checked (even if you could find a way to configure
it):

> > + t = cgroup_fork_current();
> > + while (t->css.cgroup->parent != NULL && err == 0) {

> IIRC, fork()'s error code is EAGAIN or ENOMEM. The exisiting limit of
> rlimit() returns EAGAIN.
>
> How about -EAGAIN here ? I think it's not good to add new error code for
> system calls.

EPERM seemed appropriate to me, because the administrator disallows
more than N forks. If there are practical reasons for changing it to
EAGAIN or ENOMEM, I'm ok with that. Thanks for the hint.

Max

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