Subject: Re: [PATCH 0/4] Devices accessibility control group (v2)
Posted by Sukadev Bhattiprolu on Thu, 17 Jan 2008 06:26:05 GMT
View Forum Message <> Reply to Message

```
Pavel Emelianov [xemul@openvz.org] wrote:
| sukadev@us.ibm.com wrote:
| > | > I started playing with this and noticed that even if I try to
| > | > enable read access to device [c, 1:3] it also grants access
| > | > to device [c, 1:5].
| > |
| > | Hm... I can't reproduce this:
| > |
| > | # /bin/echo 'c 1:3 r-' > /cnt/dev/0/devices.permissions
| > | # /bin/echo -n $$ > /cnt/dev/0/tasks
| > | # cat /cnt/dev/0/devices.permissions
| > | c 1:3 r-
| > | # hexdump /dev/null
| > | # hexdump /dev/zero
| > | hexdump: /dev/zero: No such device or address
| > | hexdump: /dev/zero: Bad file descriptor
| > |
| > | Maybe you have played with devs cgroups before getting this?
| > | Can you show what's the contents of the devices.permissions file
| > | in your case?
| >
| > Here is the repro again. I even tried after a reboot. Basically,
| > granting access to /dev/null is also granting access to /dev/zero.
| >
| >  # cat devices.permissions
| >  # hexdump /dev/zero
| >  hexdump: /dev/zero: No such device or address
| >  hexdump: /dev/zero: Bad file descriptor
| >  # hexdump /dev/null
| >  hexdump: /dev/null: No such device or address
| >  hexdump: /dev/null: Bad file descriptor
| >  # echo 'c 1:3 r-' > devices.permissions
| >  # hexdump /dev/null
| >  # hexdump /dev/zero
| >  0000000 0000 0000 0000 0000 0000 0000 0000 0000
| >  *
| >  ^C
| >  # cat tasks
| >  3279
| >  22266
| >  # ps
| >    PID TTY          TIME CMD
| >   3279 pts/0    00:00:00 bash
| >  22267 pts/0    00:00:00 ps
| >
|
| This all looks completely incomprehensible :(
|
| Here's my test:
| # mount -t cgroup none /cnt/dev/ -o devices
| # mkdir /cnt/dev/0
| # /bin/echo -n $$ > /cnt/dev/0/tasks
| # cat /cnt/dev/0/devices.permissions
| # hexdump /dev/zero
| hexdump: /dev/zero: No such device or address
| hexdump: /dev/zero: Bad file descriptor
```

Can you try this sequence:

- grant access to /dev/zero,
- hexdump /dev/zero
- revoke access to /dev/zero
- hexdump /dev/null
- hexdump /dev/zero.

```
| # hexdump /dev/null
| hexdump: /dev/null: No such device or address
| hexdump: /dev/null: Bad file descriptor
| # echo 'c 1:3 r-' > /cnt/dev/0/devices.permissions
| # cat /cnt/dev/0/devices.permissions
| c 1:3 r-
| # hexdump /dev/null
| # hexdump /dev/zero
| hexdump: /dev/zero: No such device or address
| hexdump: /dev/zero: Bad file descriptor
|
|
| Sukadev, could you please try to track the problem as you
| seem to be the only person who's experiencing problems
| with that.
```

I suspect the 'caching' of the last_mode (that you introduce in PATCH 2/4) combined with the fact that /dev/zero, /dev/null, /dev/kmem etc share a _SINGLE_ 'struct cdev' leads to the problem I am running into with /dev/zero and /dev/null.

Here is a what I suspect is happening (sorry, for low-level details)

Following sequence seems to repro it consistently for me:

```
 $ mount -t cgroup none /container/devs/ -o devices
 $ mkdir /container/devs/0
 $ cd !$
 cd /container/devs/0
 $ echo $$ > tasks

 $ hexdump /dev/zero
 hexdump: /dev/zero: No such device or address
 hexdump: /dev/zero: Bad file descriptor

 $ hexdump /dev/null
 hexdump: /dev/null: No such device or address
 hexdump: /dev/null: Bad file descriptor

 $ echo 'c 1:3 r-' > devices.permissions

 $ hexdump /dev/null

 $ hexdump /dev/zero
 hexdump: /dev/zero: No such device or address
 hexdump: /dev/zero: Bad file descriptor
```

No surprise so far.

```
 $ echo 'c 1:5 r-' > devices.permissions
 $ hexdump /dev/zero
 0000000 0000 0000 0000 0000 0000 0000 0000 0000
 *
 ^C
```

Now grant read access to /dev/zero and more importantly, create a properly initialized inode for it.

```
 $ echo 'c 1:5 --' > devices.permissions
```

Then remove access to /dev/zero. This removes the kobject for /dev/zero from map. Also cdev_map_reset() sets cdev->last to NULL.

```
 $ hdz
 hexdump: /dev/zero: No such device or address
 hexdump: /dev/zero: Bad file descriptor
```

Since cdev->last is NULL, chrdev_open() calls kobj_lookup() which returns a NULL kobj and the open fails.

```
 $ hexdump /dev/null # XXX
```

Again, since cdev->last is NULL, kobj_lookup() is called, this time for /dev/null. This succeeds and cdev->last is correctly initialized.

Eventually this open of /dev/null succeeds.

```
 $ hexdump /dev/zero
 0000000 0000 0000 0000 0000 0000 0000 0000 0000
```

Now the open of /dev/zero also succeeds !

I suspect that the reason is that when we first successfully read /dev/zero, we created/initialized an inode for it. This inode has the inode->i_cdev set correctly.

By reading /dev/null (marked XXX above), cdev->last is also correctly set.

But since /dev/zero and /dev/null _SHARE_ a 'struct cdev', when we call chrdev_open() for /dev/zero, we check the permissions of this common cdev and grant /dev/zero the same permissions as /dev/null.

I suspect we will get this behavior with all devices implemented by the 'mem' driver in drivers/char/mem.c. I was able to repro with /dev/full [c, 1:7])

Sukadev

____