

---

Subject: Re: [patch 2/8] allow unprivileged umount  
Posted by [Miklos Szeredi](#) on Sun, 22 Apr 2007 06:47:31 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

> > On Sat, 21 Apr 2007 10:09:42 +0200 Miklos Szeredi <miklos@szeredi.hu> wrote:  
> >  
> >> > +static bool permit\_umount(struct vfsmount \*mnt, int flags)  
> >> > +{  
> >> > >  
> >> > >  
> >> > > ...  
> >> > >  
> >> > > + return mnt->mnt\_uid == current->uid;  
> >> > > +}  
> >> >  
> >> > Yes, this seems very wrong. I'd have thought that comparing user\_struct\*'s  
> >> > would get us a heck of a lot closer to being able to support aliasing of  
> >> > UIDs between different namespaces.  
> >> >  
> >> >  
> >> OK, I'll fix this up.  
> >>  
> >> Actually an earlier version of this patch did use user\_struct's but  
> >> I'd changed it to uids, because it's simpler.  
> >  
> > OK..  
> >  
> >> I didn't think about  
> >> this being contrary to the id namespaces thing.  
> >  
> > Well I was madly assuming that when separate UID namespaces are in use, UID  
> > 42 in container A will have a different user\_struct from UID 42 in  
> > container B. I'd suggest that we provoke an opinion from Eric & co before  
> > you do work on this.  
>  
> That is what I what I have been thinking as well,

Does this mean, that containers will need this? Or that you don't know yet?

> storing a user struct on each mount point seems sane, plus it allows  
> per user mount rlimits which is major plus. Especially since we  
> seem to be doing accounting only for user mounts a per user rlimit  
> seems good.

I'm not against per-user rlimits for mounts, but I'd rather do this later...

> To get the user we should be user fs\_uid as HPA suggested.

fsuid is exclusively used for checking file permissions, which we don't do here anymore. So while it could be argued, that mount() is a filesystem operation, it is really a different sort of filesystem operation than the rest.

OTOH it wouldn't hurt to use fsuid instead of ruid...

> Finally I'm pretty certain the capability we should care about in  
> this context is CAP\_SETUID. Instead of CAP\_SYS\_ADMIN.  
>  
> If we have CAP\_SETUID we can become which ever user owns the mount,  
> and the root user in a container needs this, so he can run login  
> programs. So changing the appropriate super user checks from  
> CAP\_SYS\_ADMIN to CAP\_SETUID I think is the right thing todo.

That's a flawed logic. If you want to mount as a specific user, and you have CAP\_SETUID, then just use set\*uid() and then mount().

Changing the capability check for mount() would break the userspace ABI.

Miklos

---

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

---