

---

Subject: Re: [PATCH 2/9] security: Make capabilities relative to the user namespace.

Posted by [David Howells](#) on Wed, 23 Feb 2011 12:01:57 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

David Howells <dhowells@redhat.com> wrote:

```
> > int (*capable) (struct task_struct *tsk, const struct cred *cred,
> > - int cap, int audit);
> > + struct user_namespace *ns, int cap, int audit);
>
> Hmm... A chunk of the contents of the cred struct are user-namespaced.
> Could you add the user_namespace pointer to the cred struct and thus avoid
> passing it as an argument to other things.
```

Ah, no... Ignore that, I think I see that you do need it.

```
> +int cap_capable(struct task_struct *tsk, const struct cred *cred,
> + struct user_namespace *targ_ns, int cap, int audit)
> {
> - return cap_raised(cred->cap_effective, cap) ? 0 : -EPERM;
> + for (;;) {
> + /* The creator of the user namespace has all caps. */
> + if (targ_ns != &init_user_ns && targ_ns->creator == cred->user)
> + return 0;
```

Why is that last comment so? Why should the creating namespace sport all possible capabilities? Do you have to have all capabilities available to you to be permitted create a new user namespace?

Also, would it be worth having a separate cap\_ns\_capable()? Wouldn't most calls to cap\_capable() only be checking the caps granted in the current user namespace?

David

---

Containers mailing list

[Containers@lists.linux-foundation.org](mailto:Containers@lists.linux-foundation.org)

<https://lists.linux-foundation.org/mailman/listinfo/containers>

---