Subject: Re: design of user namespaces
Posted by ebiederm on Fri, 20 Jun 2008 19:03:21 GMT
View Forum Message <> Reply to Message

"Serge E. Hallyn" <serue@us.ibm.com> writes:
>
> Hi Eric,
>
> glad you're giving this some thought. Did you ever read over the
> approach which I outlined in May (see
> http://forum.openvz.org/index.php?t=msg&goto=30223&)? We agree on many
> points. I think we basically solve the suid problem the same way.

I hadn't read that post although I saw a part of it in your paper.
Solving that problem so unprivileged users can use a user namespace
seems to be the key to making namespaces more widely usable, and
recursive. Plus I really like the idea of a super nobody user.

> But I've moved away from a uid-to-uid mapping. Instead,
> I expand on the relation you also describe: the user who creates a user
> namespace owns the user namespace and introduce a persistant
> namespace ID.

I think there are management issues with a persistent namespace ID.
In particular: Is this user allowed to use this ID?

That is the reason I want to avoid having a generic persistent
namespace ID. Implementing a common library and then modifying the
filesystems to use it on a case by case basis sounds much more
maintainable. Then picking the wrong direction does not bind us for
all time and eternity.

> I understand that you may reflexively balk at introducing a new global
> persistant ID when we're focusing on turning everything into a
> namespace, but I think that would be a misguided reflex (like the
> ioctl->netlinke one of a few years ago). In particular, in your
> approach the identifier is the combination of the uid-to-uid mapping and
> the uids stored on the original filesystem.

Not at all. I thought I had mentioned the xattr thing as one of the
possibilities. I forgot the proper term so I probably said acls. The
practical problem is that you then have to rev your quota support. To
also support the xattr separation. In addition not every filesystem
supports xattrs. Although the common ones do.

> I do think the particular form of the ID I suggest will be unsuitable
> and we'll want something more flexible. Perhaps stick with the unsid
> for the legacy filesystems with xattr-unsid support, and let advanced

> filesystems like nfsv4, 9p, and smb use their own domain
> identifiers.

Which is why I said make it filesystem specific with support from a
generic library. No prctl just a mount option.

> But since we seem to agree on the first part - introducing a hierarchy
> between users and the namespaces they create - it sounds like the
> following patch would suit both of us. (I just started implementing my
> approach this past week in my free time). I'm not sending this as any
> sort of request for inclusion, just bc it's sitting around...

Yes. At least a loose hierarchy.

It just occurred to me that with unix domain sockets, some signals, /proc
we have user namespace crossing (child to parent) where we need to
report the uid. In that case the simple solution appears to be to use
the overflowuid and overflowgid that were defined as part of the 16bit
to 32bit transition. Otherwise it would require mapping all of the
child uids into the parent like we do with pids except using an upcall
to userspace.

Making capabilities user namespace specific if we can.

Did you have any problems with adding a user_namespace argument to
capable?

Just skimming through your patch I don't expect we will need the list
of children, and not having should reduct our locking burden.

Eric