Subject: Re: User namespaces and keys
Posted by David Howells on Wed, 23 Feb 2011 15:06:14 GMT
View Forum Message <> Reply to Message

Serge E. Hallyn <serge@hallyn.com> wrote:

> > I guess we need to look at how to mix keys and namespaces again.
>
> From strictly kernel pov, at the moment, keys are strictly usable only
> by the user in your own user namespace.

I'm not sure that's currently completely true. Key quota maintenance is namespaced, and the key's owner UID/GID belong to that namespace, so that's okay, but:

(*) key_task_permission() does not distinguish UIDs and GIDs from different namespaces.

(*) A key can be referred to by its serial number, no matter whose namespace it is in, and will yield up its given UID/GID, even if these aren't actually meaningful in your namespace.

This means request_key() can successfully upcall at the moment.

I wonder if I should make the following changes:

(1) If the key and the accessor are in different user namespaces, then skip the UID and GID comparisons in key_task_permission(). That means that to be able to access the key you'd have to possess the key and the key would have to grant you Possessor access, or the key would have to grant you Other access.

(2) If the key and someone viewing the key description are in different namespaces, then indicate that the UID and the GID are -1, irrespective of the actual values.

(3) When an upcall is attempting to instantiate a key, it is allowed to access the keys of requestor using the requestor's credentials (UID, GID, groups, security label). Ensure that this will be done in the requestor's user namespace.

Nothing should need to be done here, since search_process_keyrings() switches to the requestor's creds.

Oh, and are security labels user-namespaced?

> We may want to look at this again, but for now I think that would be a
> safe enough default. Later, we'll probably want the user creating a

> child_user_ns to allow his keys to be inherited by the child user_ns.

That depends what you mean by 'allow his keys to be inherited'. Do you mean copying all the creator's keys en mass? You may find all you need to do is to provide the new intended user with a new session keyring with a link back to the creator's session keyring.

> Though, as I type that, it seems to me that that'll just become a
> maintenance pain, and it's just plain safer to have the user re-enter
> his keys,

That would certainly be simpler.

> sharing them over a file if needed.

I'm not sure what you mean by that. Do you mean some sort of cred passing similar to that that can be done over AF_UNIX sockets with fds?

> I'm going to not consider the TPM at the moment :)

I'm not sure the TPM is that much of a problem, assuming you're just referring to its keystore capability...

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