

"Serge E. Hallyn" <serue@us.ibm.com> writes:

> Ok, the following isn't meant so much as a patch as for discussion.
> However, this may be a change we want to think about for awhile and
> collect opinions and facts. So having this file sitting in the
> kernel tree (updated with the results of any discussion we have in the
> meantime) may be useful.
>
> So what do people think? Are we ok using CAP_SYS_ADMIN? Do we
> authorize unsharing of each resource using the capability required
> to administrate the resource? Do we introduce CAP_NS_UNSHARE? Do
> we add CAP_SYS_USHARE, CAP_NET_UNSHARE, and CAP_USER_UNSHARE? Or
> do we allow unprivileged users to unshare, trusting that the actual
> administration is properly authorized?

Well if we ant to sit this in the kernel we need to remove mention
of CAP_NS_UNSHARE.

However even there the document below is only an ok first stab at
documenting things.

The big big big problem are suid executables. If we don't have suid
executables and the namespaces only apply to our children we can
unshare them all day long and no one cares. If we do have suid executables
any messing up of their context that they are not prepared to deal with
is a potential security violation.

So I think CAP_SYS_ADMIN is a good starting place. It is trivial verifiable
that it is safe. So starting there allows us to work on other aspects
of the problem for now.

I would like to remove the restrictions from creating new namespaces
however we will either have to have restrictions like the current
unprivileged mount patches, so we don't surprised root. Or we figure out
how to ensure we don't have suid applications.

Given my intuitive understanding of a complete uid namespace it
fundamentally prohibits suid executables from executing because those
users simply do not exist in the new namespace. So my hunch is we can
drop the requirement for CAP_SYS_ADMIN on namespace creation in
concert with a uid namespace creation.

So my feeling at the moment is that we need to flesh out and complete

the namespaces we have user, net, pid, user and then come back and see what we can do.

I don't think it makes sense to document a snapshot in time of the discussion with unresolved issues in the Documentation directory. Documenting what is and how we got there is fine (that is timeless) documenting what could be will likely be out of date before the patch is merged into Linus's tree.

Eric

```
> thanks,
> -serge
>
>>From 0a76e72a6900d1c47caa6aaeb5008e8408fd35e6 Mon Sep 17 00:00:00 2001
> From: sergeh@us.ibm.com <hallyn@kernel.(none)>
> Date: Fri, 9 Nov 2007 13:32:40 -0800
> Subject: [PATCH 1/1] namespaces: document unshare security implications
>
> Add a file under Documentation/namespaces to discuss security
> implications of unsharing namespaces.
>
> Signed-off-by: Serge Hallyn <serue@us.ibm.com>
> ---
> Documentation/namespaces/security.txt | 69 ++++++
> 1 files changed, 69 insertions(+), 0 deletions(-)
> create mode 100644 Documentation/namespaces/security.txt
>
> diff --git a/Documentation/namespaces/security.txt
> b/Documentation/namespaces/security.txt
> new file mode 100644
> index 0000000..c68794b
> --- /dev/null
> +++ b/Documentation/namespaces/security.txt
> @@ -0,0 +1,69 @@
> +Currently, cloning/unsharing of namespaces requires CAP_SYS_ADMIN.
> +This file addresses some ways to allow unprivileged users to
> +unshare namespaces.
> +
> +First, of course, a program unsharing namespaces can be made setuid
> +root. A slightly safer alternative a program unsharing namespaces
> +can be given cap_sys_admin in its file permitted capabilities.
> +
> +Requiring CAP_SYS_ADMIN is a legacy behavior stemming from the
> +fact that the original namespace, the mounts namespace, required
> +CAP_SYS_ADMIN for cloning. Unfortunately CAP_SYS_ADMIN is used
```

- > +to authorize many other actions, so that giving away CAP_SYS_ADMIN
- > +to allow unsharing also allows the unsharing user many other
- > +privileges.
- > +
- > +Instead of using CAP_SYS_ADMIN, a new capability, CAP_NS_UNSHARE,
- > +could be introduced. This way a program or user wouldn't have
- > +to be granted full CAP_SYS_ADMIN rights to be able to clone/unshare
- > +namespaces, but a fully unprivileged user still could not
- > +clone/unshare.
- > +
- > +Or, unsharing namespaces could be turned into an entirely unprivileged
- > +operation. Unsharing a namespace does not give the user any new
- > +rights to modify the unshared resource in the new namespace. For
- > +instance, after doing
- > + unshare(CLONE_NEWNS)
- > +the unprivileged task can't perform any mount actions he couldn't
- > +before the unshare.
- > +
- > +Is it safe to allow namespace unsharing by nonprivileged users?
- > +The following tries to answer that question per-namespace:
- > +
- > +UTS
- > + Safe. Can't sethostname without cap_sys_admin.
- > + But similarly, since you can't sethostname without
- > + cap_sys_admin there may not be any point in
- > + unsharing your utsns without cap_sys_admin.

Yes. But you can hide from a future sethostname. So because the your hostname won't change the effect is as if you had set it.

- > +IPC
- > + could be unsafe if a setuid root application expects
- > + to talk to a privileged server in the init ipc ns.
- > + The opencryptoki pkcs11d might be an example.
- > +VFS
- > + user might hide himself from root mount activity,
- > + but in general vfs ns are safe and don't change the
- > + safety of user mounts.
- > + Since there is mount activity (and more to come) that
- > + users can do without CAP_SYS_ADMIN, it may be useful
- > + to allow unshare(CLONE_NEWNS) without CAP_SYS_ADMIN.

I assume you mean the mount namespace. There is still the stale state problem and suid executables.

- > +PID
- > + safe

Nope. It can hide processes that a suid executable would like to signal... It messes up the use of pid files and the like.

> +User

> + user can get around quotas. As user namespaces are
> + fleshed out, root in a user namespace will be confined,
> + and equivalent user ids between namespaces will be
> + isolated.

> +Net

> + user won't have cap_net_admin so won't be able to set
> + up networking in new ns. Biggest risk is due to
> + any root services which don't handle failure due to
> + no network right.
> + If netlink isn't handled right, user might be able to
> + get around the audit daemon! That's very bad.
> + Since network has it's own CAP_NET_ADMIN, it may make
> + sense to require that to unshare(CLONE_NEWNET). But
> + requiring different capabilities to unshare different
> + resources may be too confusing/annoying.

> +

> +All these share the threat of extra memory consumption, but
> +this can be addressed using cgroups.

> --

> 1.5.1

>

>

> Containers mailing list

> Containers@lists.linux-foundation.org

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

Containers mailing list

Containers@lists.linux-foundation.org

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