Subject: Re: [PATCH 1/2] namespaces: introduce sys_hijack (v10)
Posted by ebiederm on Fri, 30 Nov 2007 22:09:28 GMT
View Forum Message <> Reply to Message

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

> Quoting Eric W. Biederman (ebiederm@xmission.com):
>> Mark Nelson <markn@au1.ibm.com> writes:
>>
>> > Hi Paul and Eric,
>> >
>> > Do you guys have any objections to dropping the hijack_pid() and
>> > hijack_cgroup() parts of sys_hijack, leaving just hijack_ns() (see
>> > below for discussion)?
>>
>> I need to step back and study what is being proposed.
>>
>> My gut feeling is that you are proposing something that does not
>> support forking me a process inside a container so I can have a
>> shell without having to run a login program.
>
> Hmm, depends on exactly what you want, but you may be right.
>
> In terms of namespaces it'll be in the target container, including
> having a pid in the container.

Yes, which is generally what you want for a magic login shell.

> The most dangerous part about the purely ptrace method you mention is
> that pieces of the ptraced process' environment may leak, pollute,
> and attack your new process. But it shouldn't be impossible to do
> it safely. Just tedious.

Yes. It is that use case more then anything I am concerned with.

>> There is a reason I proposed ptrace as an initial prototype.
>>
>> All of the other uses of enter in a namespace context I feel confident
>> we can support by just having proper virtual filesystems available
>> to processes outside of the container. For monitoring and control.
>
> I think you're showing an unhealthy amount of trust in both our ability
> to provide full fs-based controls to all filesystems and to your own and
> other people's abilities to never mess up a container. As an example of
> the former, will you be able to create and configure a network interface
> or add iptables rules purely through fs interface?

Well the fs interface for monitoring is pretty much on target.
As for iptables just get me a proper socket outside of the container
and I can control things. (Pity we can't do plan 9 style binds of file
descriptors the mount namespace).

> As an example of the
> latter, one little mistake and your container's mounts ns may no longer
> be a slave of yours or of /containers/c_22/root. It might take you
> years to figure out that all the time when you were doing
>
> mount --bind /mnt/nas /containers/c_22/root/mnt/backup
> echo 1 > /containers/c_22/root/root/backup-trigger
> read /containers/c_22/root/root/backup-callback
> umount /containers/c_22/root/mnt/backup
>
> your backups weren't going to your network storage but just being copied
> on local disk...

Yes, that could be nasty.

> BUT more importantly, it sounds like you are not interested in
> hijack_pid or hijack_cgroup, and Paul is only intersted in
> hijack_ns. So noone will mind if we dump the other two? It
> should greatly simplify the patch!

I don't expect so. So far filesystem and file descriptor based
interfaces I am confident that we can use outside of a container
(which really is most of everything), with our current infrastructure.

Doing it that way seems to provide more natural access controls.

So I am mostly interested in some way to get a magic login shell
inside a chroot with a filedescriptor that I have passed for
my input and output. Make it a unix domain socket and I can
pass all of the filedescriptors I want in out of the little world.

I like the concept of using something like sys_hijack for that,
rather then ptrace, it can be a lot less of a hack.

I will come back to this and look a bit more once we have the pid
and network namespaces in decent shape. Thanks for keeping the
idea alive.

Eric

____
Containers mailing list
Containers@lists.linux-foundation.org

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