Subject: Re: [RFC] ns containers (v2): namespace entering
Posted by ebiederm on Tue, 20 Feb 2007 17:38:12 GMT
View Forum Message <> Reply to Message

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

> Quoting Eric W. Biederman (ebiederm@xmission.com):
>>
>> You miss an issue here. One of the dangers of enter is leaking
>> capabilities into a contained set of processes. Once you show up in
>
> Good point. As wrong as it feels to me to use ptrace for this, the
> advantage is that none of my task attributes leak into the target
> namespace, and that's a very good thing.
>
> I do wonder how you specify what the forced clone should run.
> Presumably you want to run something not in the target container.
> I suppose we can pass the fd over a socket or something.

Yes. At least in the case without a network namespace I can setup
a unix domain socket and pass file descriptors around. I think my solution
to the network namespace case was to just setup a unix domain socket in
the parent namespace and leave it open in init. Not a real solution :(

> Herbert, does this sound like it suffices to you? Of course, it does
> mean that you cannot switch just a few namespaces. How does that limit
> you?
>
>> /proc processes can change into your working directory which is
>> outside of the container for example.
>
> I suppose I could get into specific ways that this could be prevented
> from being exploited, but for now I'll just stick to agreeing that there
> would be a whole bunch of issues like this, and we'd likely miss more
> than one.

Thanks. I think what we want is an API where what is leaked is an
explicit decision rather then everything leaking implicitly and the
user space programmer has to run around and close all of the holes.

>> Yes. This looks to be the most sensible thing and now that we have
>> struct pid we don't have to special case anything to implement a
>> pid showing up in multiple pid namespaces. So it is my expectation
>> that each process will show up in the pid namespace of all of it's
>> parents up to init.
>
> In fact we were thinking there could be an additional clone flag when
> doing CLONE_PIDNS to determine whether all processes get pids in all

> ancestor pid_namespaces or not, since really the only change should be
> an overhead at clone() for allocing and linking the additional struct
> pid_nr's.

I doubt it is worth the hassle of making the code conditional.

>> > Or, if it is ok for the pid namespace operations to be as coarse as
>> > "kill all processes in /vserver1", then that was going to be implemented
>> > using the namespace container subsystem as:
>> >
>> > rm -rf /container_ns/vserver1
>>
>> To some extent I have an issue with this since we have kill and
>> signals and other mechanisms already existing for most of this
>> the duplication at the very least seems silly.
>
> But you don't really. If I want to kill all processes in a child
> container, I don't have an easy way to list only the processes in the
> child container using, say, ps. Those processes are just lumped in with
> my own, and with those from all my other child containers.

True. Mostly because that is the interface that makes the most
sense most of the time.

> So we'd end up looping over all pids, checking their ->container, then
> killing them, and hoping that nothing funky happens meanwhile like that
> process dies and pids wrap around while i'm sleeping (ok, unlikely, but
> not impossible).
>
> I'm not saying I'm hung up on doing it with an rm /container_ns/vs1. If
> the only use for the ns container subsystem ends up being to tie
> resource management to containers, I'll be very happy.

We've got to get a better name for those things....

Regardless one of the requirements of the pid namespace is that we have
a process grouping of the entire namespace that kill(-1 can signal
and there is no reason not to have that process grouping available on
the outside of the pid namespace as well.

Eric

_____