Subject: Re: [RFC] ns containers (v2): namespace entering
Posted by ebiederm on Tue, 20 Feb 2007 03:08:45 GMT
View Forum Message <> Reply to Message

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

> (Updated patchset addressing Paul's comments. Still looking more
> for discussion on functionality, but barring much of that I guess
> I'll do something with tsk->fs then send the set to lkml)
>
> The following patchset uses the ns container subsystem to implement
> namespace entering. It applies over the container patchset Paul
> sent out earlier today.
>

Let's describe the problem. There are two usage scenarios.
- Working with the parameters of an existing namespace.
  (For example: changing or reading the hostname of a container you are not
   in, or setting/reading various resource limits).
- Logging in to an existing namespace. I.e. You are the uber sysadmin
  and the person running a container has a problem or something similar.

Until we work through the details of the pid namespace there are going
to be details of this we cannot finalize, because we will not have finalized
how traditional process groups work. Something more for my todo guess.

I have given this some though and I can describe how far we can go without
implementing a traditional enter, which is a very long ways.

For the general management functions most things already have or can
have added a filesystem interface, and we just need to make that
filesystem interface per process...

That is we can do like what is currently done with /proc/mounts and
make /proc/sys a symlink to /proc/self/sys and /proc/<otherpid>/sys
can be that other processes view of all of the sysctls.

We can do the same thing with /proc/net, and /proc/sysvipc and
possibly /sys/net although that is a more difficult. sysfs is a pain
to work with.

For the actual enter functionality what is currently possible is ugly
to implement but extremely close to something useful. You can use
sys_ptrace and pick a random process and with a little care cause

it to fork and exec the executable of your choice. With these
manipulations you can create things like unix domain sockets and
pipes and that can be used to talk with the parent process. It has
been a while since I have done this so I don't remember the exact
limits are but I have gotten a fully functional login shell this way.
The big practical problem was who is the parent process.
CAP_SYS_PTRACE is the governing capability here.
I've got the code around someplace for doing this if you are curious.

If we optimize/cleanup this case it looks a lot like what (I think it
was Cedric) was proposing about a year ago. A magic fork that does
the enter for you. Since the caller controls the binary we don't
have the usual concerns about the magic fork becoming spawn, because
we can program the binary to do anything else it needs after the fork.

> This is RFC not just on implementation, but also on whether to do
> it at all. If so, then for all namespace, or only some? And if not,
> how to facilitate virtual server management.

My gut feeling is the best way to go is something that is a refinement
of the two techniques I have listed above.

>
> = Security =
>
> Currently to enter a namespace, you must have CAP_SYS_ADMIN, and must
> be entering a container which is an immediate child of your current
> container. So from the root container you could enter container /vserver1,
> but from container /vserver1 you could not enter /vserver2 or the root
> container.
>
> This may turn out to be sufficient. If not, then LSM hooks should be
> added for namespace management. Four hooks for nsproxy management (create,
> compose, may_enter, and enter), as well as some security_ns_clone hook for
> each separate namespace, so that the nsproxy enter and compose hooks have
> the information they need to properly authorize.

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
/proc processes can change into your working directory which is
outside of the container for example.

> = Quick question =
>
> Is it deemed ok to allow entering an existing namespace?
>
> If so, the next section can be disregarded. Assuming not, the following
> will need to be worked out.

I think we need to take a good hard look at the alternatives, because
the are very functional.

> = Management alternatives =
>
> Mounts
>
> Ram has suggested that for mounts, instead of implementing namespace
> entering, the example from the OLS Shared Subtree paper could be
> used, as follows (quoting from Ram):

Does anyone know why we have the shared subtree entering instead of
a enter on a fs namespace? I'm curious about the reasoning for that
design decision.

> Herbert, and anyone else who wants mounts namespace entering, is the
> above an acceptable alternative?
>
> net+pid+uts
>
> Not sure about uts, but I'm pretty sure the vserver folks want the ability
> to enter another existing network namespace, and both vserver and openvz
> have asked for the ability to enter pid namespaces.
>
> The pid namespaces could be solved by always generating as many pids for
> a process as it has parent pid_namespaces. So if I'm in /vserver1, with
> one pid_namespace above me, not only my init process has an entry in the
> root pid_namespace (as I think has been suggested), but all my children
> will also continue to have pids in the root pid_namespace.

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.

> 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.

> Any other (a) requirements, (b) ideas for alternate pid and network

> ns management without allowing namespace enters?

See above.

I'm stretched pretty thin at the moment, so this will have to do for
a first set of comments.

Eric

_____
Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers