Subject: Re: Two newbie questions on containers
Posted by Rob Landley on Wed, 12 Jan 2011 17:35:41 GMT
View Forum Message <> Reply to Message

On 01/11/2011 04:08 PM, Timur Tabi wrote:
> Hi,
>
> I'm in the process of learning about Linux containers, including cgroups, and
> the learning curve seems pretty steep to me.

Join the club. I need to write up documentation on what I'm learning...

> So I have a couple newbie
> questions for all of you. Any detailed answered are greatly appreciated.

Container support consists of a number of things. The cgroups
filesystem is one, the napespace flags to clone (all the ones starting
with CLONE_NEW*) are another, various synthetic filesystems like devpts
have "-o newinstance". And of course it's all built on top of chroot.

The LXC userspace tool attempts to tie all of these together into
something coherent. They have their own mailing list, off of lxc.sf.net.

I wrote up my ignorance at:

  http://landley.livejournal.com/47024.html
  http://landley.livejournal.com/47205.html

> 1) For the PowerPC architecture, is there anything that is "missing"? I can't
> really tell how much of cgroups and lxc is architecture-specific, and there
> appears to be PowerPC support for both already. I'd like to know if this
> another one of those areas, like KVM, where x86 is fully implemented and PowerPC
> support is lagging.

Containers support is basicaly chroot on steroids. It attempts to build
_up_ from chroot to provide efficient fully isolated virtual systems,
the same way paravirtualization is attempting to strip down
virtualization to reinvent the microkernel. Both approaches have their
fundamental limits: containers are never going to boot Windows and
paravirtualization is unlikely to scale much better than Multix or "The
Hurd".

But a big advantage of containers is it's about as portable as chroot.
It hasn't received a lot of testing on other targets, but there's no
fundamental reason it shouldn't work just fine.

This might help:

http://lxc.sourceforge.net/index.php/about/kernel-namespaces /

> 2) Given a random device driver, like a driver for a serial port, is there an
> opportunity for the driver to be enhanced to support cgroups or lxc?

Define "support".

There's two main categories of device containerization:

1) Selective visibility, so you can move a physical device into a container and have it _only_ show up there, and not be visible elsewhere.

2) Synthetic devices, such as /dev/console in a container that a host LXC instance can attach to and be at the other end of. (TUN/TAP ethernet interfaces are another example.) These generally exist to let the container talk to the outside world with host-controllable routing.

However I'm assured that "fully transparent" containers are not a goal. So having /dev/console be a pty if necessary may be "good enough" for a given deployment.

(Also, note that since a container inherits a filesystem via chroot (with whatever shared subtree mount splices the host cares to set up before chrooting), it has less need for block device access than usual.)

> Doing a
> simple search of the kernel source code, I don't really see any drivers making
> calls into any cgroup code, so I don't understand how to restrict device access
> to a specific container or cgroup.

I'm banging on the CONFIG_NET_NS stuff a bit, although I'm sure there's plenty of bugs for all. :)

Note that there are longstanding out-of-tree containerization solutions (most notably openvz) which are implemented differently (new syscalls, an approach that got vetoed) than the containers support that made it into the kernel (Google's submission, based on something SGI did).

Those out of tree things do stuff that Linus's tree still doesn't. They're porting stuff to the new way of doing things and submitting it upstream, but there's still a lot of shoveling left to do. So you may have heard of capabilities that simply aren't in mainline yet.

Rob

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