Subject: Re: OverlayFS (was Re: CUDA support inside containers)
Posted by abufrejoval on Tue, 22 Nov 2016 08:42:43 GMT
View Forum Message <> Reply to Message

khorenko wrote on Mon, 21 November 2016 17:11abufrejoval wrote on Mon, 21 November 2016 15:18I guess OverlayFS is what I need to selectively replicate /proc and /sys contents from the host to make the CUDA runtime happy.

I remember reading about it on lwn.net and it looks like it's actually been backported from 3.18 to RHEL/CentOS/(OpenVZ?)7 3.10 kernels to support Docker, but it seems it's kernel only and missing support from the userland tools ("mount: unknown file system type 'overlayfs'...").

It's quite maddenning, because device access to /dev/nvidia* from inside the container in general seems to be supported in OpenVZ: I get a 'proper' "invalid argument" when doing a 'cat /dev/nvidia-uvm' and not the dreaded "permission denied" I get from LXC on the native CentOS.

You can use overlayfs inside a Virtuozzo 7/OpenVZ 7 Container, all you need is to load the appropriate kernel module on the host.

Did you try to get further with mounting /proc/modules inside a Container with the file with content from the host's /proc/modules?

I write too much, that's why the answers got lost

Yes, I did try mounting /proc/modules via a copied file from the host which I then bind mounted as you suggested.

And then the runtime library just wanted the *next* file which wasn't there. I copied that, too but eventually I got stuck at /proc/devices, which is a *directory* on the host but an empty *file* on the guest: I couldn't bind mount a directory over the empty file (nor could I delete the empty file from the guest's procfs).

That's why I thought I might potentially get there using the overlayfs, which really seems to support all kinds of dirty tricks.

And there the trouble is, that the Virtuozzo 3.10. kernel supports the overlayfs functionality via its sys-call interface thanks to Docker running so much quicker with it. But the actual user land tool 'mount' doesn't understand the -t overlayfs parameter: I'd have to go and get one from e.g. a more up-to-date Fedora and statically compile that against a matching c-library etc.

In short words: All kinds of trouble when ideally Nvidia should offer a run-time library option, which doesn't do all these 'convenience' checks.

I've sent a request to Nvidia accordingly and I'm hoping for them to fix the issue at the source.

Of course somewhere within Virtuozzo there must be a table which decides which elements in /proc and /sys are visible to guests and which need translation (e.g. UID or PID mapping).

I should be able to patch that code and build a 'matching' CUDA kernel, just to see if that eventually solves the problem, too.

But I'd invest that effort only, if I could be sure that CUDA enabled Docker workloads also run on both the host and inside OpenVZ containers, because that would make OpenVZ feature complete with regards to the environment I need to build. That requires support for the current docker-engine 1.12.1 on both sides and evidently Docker and OpenVZ don't get along as well as I had hoped any more. Some tests using older Docker variants had looked rather promising early this year, but Nvidia has built a docker-plugin, which requires 1.10 or greater.

Essentially I want to support two major 'client' workloads: CUDA enabled Docker images and CUDA enabled 'IaaS' container.
Ubuntu delivery both, but with--well Ubuntu and LXC both of which require significant relearning and additional risks.

I really don't want to go down that road, but at the moment I have no choice.
