Subject: CUDA support inside containers
Posted by abufrejoval on Sat, 12 Nov 2016 13:33:32 GMT
View Forum Message <> Reply to Message

I'm trying to set up a playground for machine learning engineers based on a beefy dual socket Xeon E5 with plenty of RAM and a Tesla GPU from Nvidia (pure compute GPU, no video-out).

CUDA seems to manage multi-tasking well enough, as long as limited resources such as GPU RAM are not exhausted, multiple applications run happily side-by-side.

The main issue I'm trying to solve is that most machine learning application stacks come with distinct userlands: There is Ubuntu, CentOS or plain Docker in all kinds of versions and I'd like them to co-exist happily without re-installation or exclusivity (PCI-passthrough to VM) and that's after all what container virtualization was designed to do, right?

While I've seen reports that with Docker CUDA workloads are possible, I'd always rather run Docker inside an OpenVZ container and I'd also rather give the guys the IaaS experience they are used to. They are also likely to do development work inside there and that's where Docker starts to become cumbersome.

Problem is that this new generic system resource, the GPU, today isn't quite treated like CPU, RAM or storage by OpenVZ: There is no built-in redirection layer for GPUs (BTW: How would that look with AMD APUs?).

The CUDA software evidently needs access to /dev/nvidia* to get things done and inside a container that currently seems a no-no.

Since this is a rather generic issue going forward: Any ideas how you'll want to implement that?

And is there a dirty hack which could be done to make this possible in the mean-time?

Security isn't an issue in this context: They are all friends in this case. But of course, security and strict resource allocation would be required for the production variant going forward.