Shifter experiences with high performance containers

Kean Mariotti (kean.mariotti@cscs.ch), Lucas Benedicic, Alberto Madonna, Felipe A. Cruz. CSCS, Swiss National Supercomputing Centre.

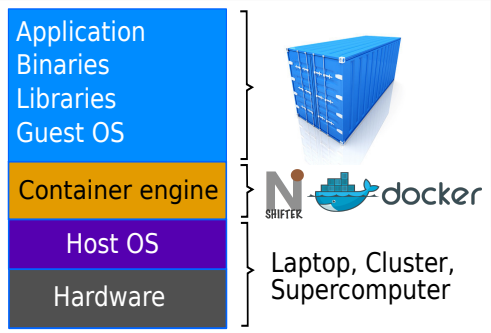
Abstract

High Performance Computing (HPC) applications have to reliably run across many platforms and environments. Containers are a type of lightweight virtualization technology that attempts to solve this problem by packaging applications and their runtime environments into standard units of software that are: portable, easy to build and deploy, have a small footprint and a low runtime overhead. In this poster, we present a container workflow for HPC applications using two tools: Docker, for building and packaging applications into containers; and Shifter, a container runtime that has been built to fulfill the specific needs of HPC. We will present use cases of building and testing containers on workstations and deploying them on HPC systems, where they take advantage of the available specialized-hardware: fast interconnects and GPU accelerators. We will provide performance results for a variety of scientific applications and discuss how such workflow can aid HPC users.

1. Building and maintaining HPC applications is hard

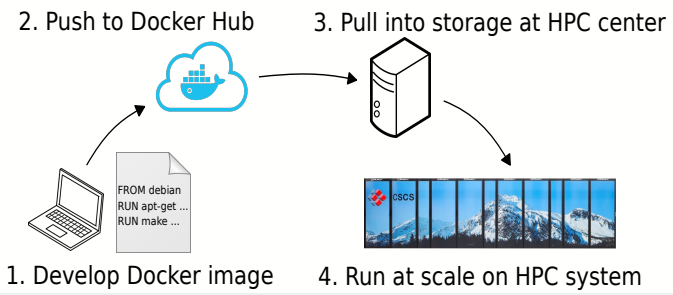
- Applications are built directly on HPC system
- Dependencies are often built from source
- Slow modify-build-test cycle
- The application is not portable
- Build process often not reproducible

2. Virtualization containers to the rescue



- Docker: mature, widely used container ecosystem
- Docker Hub provides many officially supported images out of the box (TensorFlow, Trilinos, CUDA Toolkit, etc.)

3. Workflow



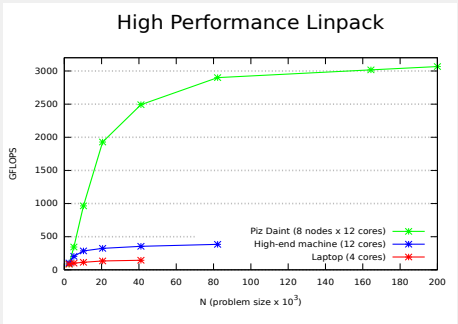
4. Shifter [1][2]

- Container engine for HPC
- Designed for performance
- Integration with workload manager
- Compatible with Docker

5. Shifter + Docker

- Applications are built on the workstation
- Dependencies are installed from package manager
- Rapid modify-build-test cycle
- The container is portable
- Enforced reproducibility of build process
- Same performance as native application

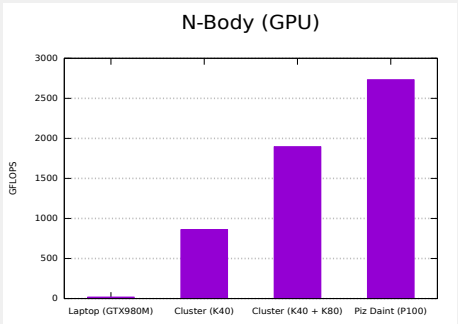
6. Examples of applications



Results of High Performance Linpack (HPL). The lines show the results across different systems. The vertical axis shows the performance in GFLOPS. The horizontal axis shows the different problem sizes.

The same container image can take advantage of the CPU power of each system showcasing good scalability.

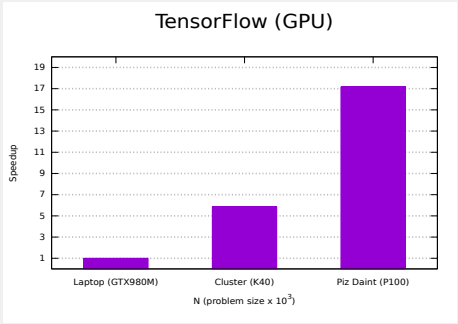
On Daint, the application also uses the high-speed interconnect for inter-node communications.



Results of the CUDA N-Body benchmark with N=200'000. The bars show the results across different systems in GFLOPS.

The performance difference between native and container is within 0.4%.

The same container image can take advantage of different GPU hardware (also multiple GPUs).



Results of MNIST example (handwritten digit recognition) from the TensorFlow tutorials. The bars show speedups across different systems (Laptop is the baseline).

The official TensorFlow Docker image (released weekly) works out of the box on different systems.

7. Specialized Hardware support

- Container accesses GPU
- Container accesses native MPI and high-speed interconnect
- Same performance as native application [3]

8. Conclusions

- Docker: faster build, portable and reproducible
- Shifter: run containers at scale with native performance
- Shifter + Docker: simplified workflow without performance penalties

References

[1] D. M. Jacobsen and R. S. Canon, "Shifter: Containers for HPC". In Proceedings of the Cray Users Group Conference (CUG'16), 2016.
[2] L. Benedicic, M. Gila, S. Alam, and T. Schulthess, "Opportunities for container environments on Cray XC30 with GPU devices", in Cray Users Group Conference (CUG'16), 2016.
[3] L. Benedicic, F. A. Cruz, A. Madonna, K. Mariotti, "Portable, high-performance containers for HPC", eprint arXiv:1704.03383