Evaluation of containers as a virtualisation alternative for HEP workloads

Gareth Roy¹, Andrew Washbrook², David Crooks¹, Gordon Stewart¹, Gang Qin¹, Samuel Skipsey¹, Dave Britton¹

¹University of Glasgow ²University of Edinburgh
Outline

- What are Containers?
- The Container ecosystem
- Container deployment and management
- HEP benchmarking
- Conclusions
Containerisation is a form of **OS level virtualisation**

- The Linux kernel hosts multiple partitioned user-land instances (Virtual Environments)
- Accomplished through separate namespaces for filesystem mounts, network, processes and users
- Backing storage can be Copy-on-Write or a union filesystem (UnionFS/AUFS)

**Linux Container**

Containers Timeline

- 1982: BSD Chroot
- 2000: FreeBSD Jails
- 2004: Solaris Zones
- 2005: OpenVZ
- 2006: Process Containers
- 2007: Cgroups
- 2008: LXC
- 2013: Docker
- 2014: LMCTFY
- 2014: Rocket
Comparison with Virtual Machines

**Virtual Machine**

- **Pros:**
  - OS independent
  - Security Model
  - Live migration
  - Mature ecosystem

- **Cons:**
  - Full system image
  - Slow startup and build
  - Memory consumption
  - Opaque to host

**Linux Container**

- **Pros:**
  - Low barrier of entry
  - Fast Instantiation
  - Native Performance
  - Deployment Flexibility

- **Cons:**
  - Restricted to Linux
  - Shared Kernel
  - Security Model
  - Young Ecosystem
The Container Ecosystem

- There is a growing ecosystem of tools and services to deploy, manage and orchestrate containers.
- The most popular container application platform is **Docker**.

**LXC** - tools for container lifecycle management.

Application virtualisation engine based on containers (**LXC, libcontainer**).

**Kubernetes** - Docker container orchestration system for large scale application deployment.

Open source minimal OS specifically designed to host and cluster application containers.

**Core OS**

Tools for working with containers focusing on the *Application Container Image* (an open standard for container formats).
Openstack Container Management

- Explored the readiness of OpenStack to natively support the management of containers
- Enables easy integration with cloud infrastructure available in WLCG
- A Docker driver is not in the current Openstack release (Juno)
  - Manually install driver: https://wiki.openstack.org/wiki/Docker

```bash
# docker pull cern/sl6-lite
# docker save cern/sl6-lite | glance image-create --is-public=True --container-format=docker --disk-format=raw --name cern/sl6-lite
```
HEP Containers

Container comparison with CERNVM image

- Pull base CentOS 6 image from Docker registry
- CVMFS mounted as an external volume
- Increase storage in the container for datasets and job output

CVMFS integration

- CVMFS requires root-privileges for FUSE interaction.
  
  *Either run a privileged container and export the CVMFS volume to other containers*
  
  - Security implications
  
  *Or export the CVMFS volume from the host*
  
  - Not a flexible hypervisor solution
  
  - Can lead to issues with other container management tools (CoreOS, Project Atomic)
HEP Workload Performance Testing

Motivation

- Do containers offer native performance for realistic HEP-based workload?
- What is the performance penalty for using Virtual Machines over bare metal (and containers)?

Workload types

- HEPSPEC benchmark
- Geant4 Monte Carlo Simulation
- Event Reconstruction
- Monte Carlo event generation
HEP Workload Performance Testing (2)

**Test Platform**

- Run each HEP workload type on two testbed servers; one containing an Intel Xeon processor and the other an Avoton processor
  - **Avoton**: Low power Atom-based 22nm SoC device
- Run each workload type on bare metal, in a virtual machine (KVM) and in a container (Docker)
- Adapted a µCERNVM image for testing purposes
- Tested both a RAW image file and a LVM partition as VM backing storage

**Test Patterns**

- Run each test multiple times to validate performance and timing consistency
- Run single core and \( N (= \text{number of cores}) \) simultaneous workload instances
Containers are within 1% of “native” HEPSPEC performance

Benchmark score for VMs are 14.7% less for the Xeon and 15.3% less for the Avoton compared to the native HEPSPEC (64-bit) score
MC Simulation Results

- Focus on relative performance in event processing loop
- CPU time/event (as reported by application) is averaged over 40 events

- Containers demonstrate near-native performance
- Single process MC simulation timing performance is 13.4% lower for the Xeon and 26.4% lower for the Avoton
- For 16 (8) simultaneous processes: VM is 24.7% lower for Xeon (32.5% lower for Avoton)
Event Generation and Reconstruction Results

Event Generation

- Measured total time taken to generate large sample of $Z\mu\mu$ events
- VMS lose only 5% performance compared to bare metal and containers

Event Reconstruction

- CPU time/event averaged over 50 events
- VMs show 22.6% performance drop for a single process, 18.8% for 8 simultaneous processes
Conclusions

- Containers are a compelling alternative to whole system virtualisation
- The performance of containers for HEP workloads are similar (if not the same) as native execution
- Virtual Machines observed to give a reduction in performance of 15-20% on HEP workloads
  - Caveat: VMs could be tuned to give better performance

Future Work

- Potential deployment of containers on existing HEP cloud resources
- Does HEP lend itself to the single application model preferred by Docker?
  - Consider: middleware deployment, distributed computing components (e.g pilots)
- Effort has already started (see next presentation!)
Any Questions?