HEP reference workloads in containers: a proof of concept

D. Giordano (CERN)

HEPiX Benchmarking Working Group
Friday 19 May 2017

Requirements for HEP benchmark(s)

- Scale, within a given accuracy, with a representative WLCG job mix
- Target architectures/OS/infrastructures adopted and to be adopted in WLCG

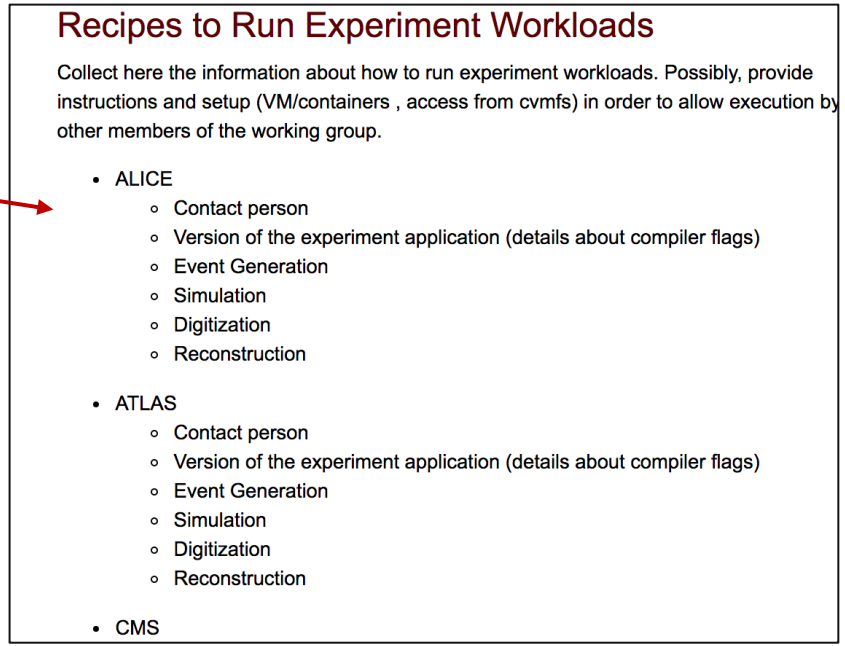
In view of a HS06 successor

Prepare the test environment

- Disentangle effects such as
  - Bare-metal server Vs VMs, HT ON/OFF, different OS, job variability, load on neighbor slots,…
- Identify testbed with representative set of hardware models
  - Include Non-default hardware models (e.g. ARM) of interest in HEP
- Perform reproducible studies
  - **Document procedures**
  - Experiments: **identify and share representative job types** (e.g. via CVMFS, containers, etc..)
  - Have access to monitoring data of production jobs

The WG needs from Experiments

- Candles: your selected workloads
  - See form to fill in the WG twiki
    - Action still pending
- Tools: Expose an easy approach to run applications even without knowledge of the experiment setup
  - The Candles will not change
    - Same executable version, same conditions, same event sequence
  - Site admins happy to run something that doesn't require additional (complex) configurations
    - Eg. additional cvmfs areas, complex scripts



Recipes to Run Experiment Workloads

Collect here the information about how to run experiment workloads. Possibly, provide instructions and setup (VM/containers , access from cvmfs) in order to allow execution by other members of the working group.

- ALICE
  - Contact person
  - Version of the experiment application (details about compiler flags)
  - Event Generation
  - Simulation
  - Digitization
  - Reconstruction
- ATLAS
  - Contact person
  - Version of the experiment application (details about compiler flags)
  - Event Generation
  - Simulation
  - Digitization
  - Reconstruction
- CMS

https://twiki.cern.ch/twiki/bin/view/HEPIX/CpuBenchmark#Recipes_to_Run_Experiment_Worklo

For example

- To run KV few steps are needed
  - Assumed there is cvmfs mounted and SLC6 (with enough standard packages)
  - Configuration of the environment

```
export ATLAS_LOCAL_ROOT_BASE="/cvmfs/atlas.cern.ch/repo/ATLASLocalRootBase"
source ${ATLAS_LOCAL_ROOT_BASE}/user/atlasLocalSetup.sh -q
source $AtlasSetup/scripts/asetup.sh 17.8.0.9,AtlasProduction
```

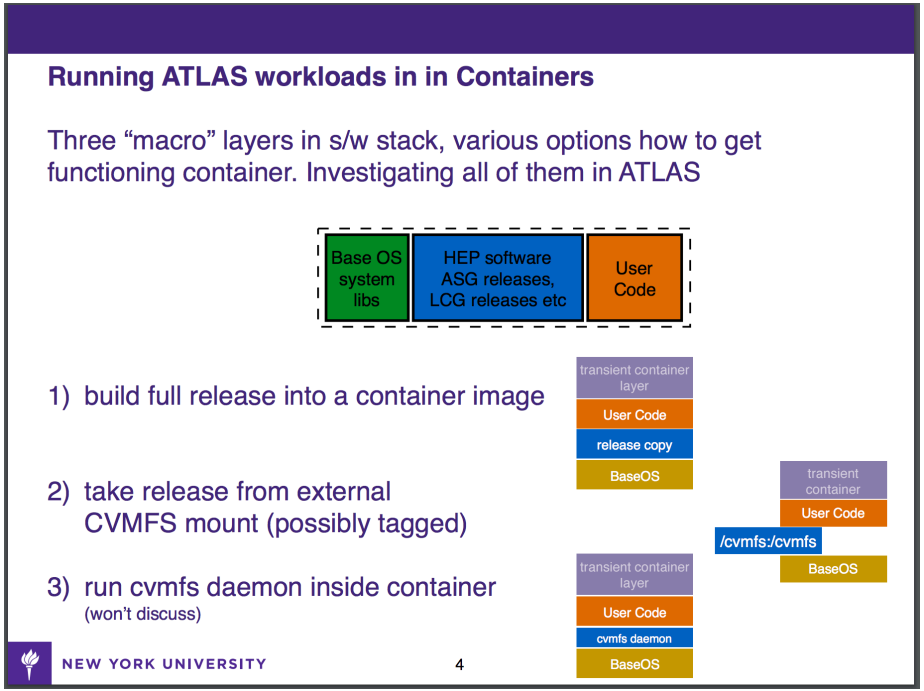
  - Run the athena application

```
    /cvmfs/atlas.cern.ch/repo/sw/software/x86_64-slc6-gcc46-
opt/17.8.0/AtlasProduction/17.8.0.9/InstallArea/share/bin/AtlasG4_trf.py
outputHitsFile='KitValidation-SimulHITS-17.8.0.9.pool.root' maxEvents='100' skipEvents='0'
preInclude='KitValidation/kv_reflex.py,preInclude.SingleMuonGenerator.py'
geometryVersion='ATLAS-GEO-16-00-00' conditionsTag='OFLCOND-SDR-BS7T-04-03' 2>&1 >out_$1.log
```

- Documentation is crucial
  - To know how to setup the environment, the application parameters, and the proper configuration to run a given workload

Container activity in ATLAS

- Atlas activity triggered interest in the container approach
  - Some drawbacks
    - Large images or cvmfs needed
- Possible solution
  - Suggested by D. Abdurachmanov (CMS)
  - Snapshot only the cvmfs files that are needed
  - Using this toolkit
    - CARE: Comprehensive Archiver for Reproducible Execution
      - Based on ptrace
    - https://github.com/proot-me/PRoot



Lukas Alexander Heinrich

Example: slim KV benchmark

- Create a Docker image based on slc6-base and that contains
  - Only libraries needed to run athena
    - Limited set of files from cvmfs (**624MB**)
      - **atlas-condb.cern.ch atlas-nightlies.cern.ch atlas.cern.ch sft.cern.ch**
    - A number of standard applications
  - Slim Conditions sqlite file (thanks to L. Rinaldi)
    - ALLP200-OFLCOND-SDR-BS7T-04-03.slim.sqlite (**490KB**)
  - Total size of the container **1.16 GB**

```
FROM gitlab-registry.cern.ch/linuxsupport/slc6-base:latest

MAINTAINER Domenico Giordano <domenico.giordano@cern.ch>

RUN yum install -y \
    which \
    man \
    file \
    util-linux \
    gcc \
    wget \
    tar \
    perl ; yum clean all
#workaround https://github.com/CentOS/sig-cloud-instance-
images/issues/15

RUN wget https://test-giordano.web.cern.ch/test-
giordano/Benchmarking/cvmfs_kv-bmk-v17.8.0.9.tgz; tar -xvzf
cvmfs_kv-bmk-v17.8.0.9.tgz; rm cvmfs_kv-bmk-v17.8.0.9.tgz

COPY ./kv-bmk /kv-bmk

ENTRYPOINT ["/kv-bmk/kv-bmk.sh"]
CMD ["-n0"]
```

Details

- How to run it
  - `docker run -it --rm gitlab-registry.cern.ch/giordano/hep-workloads:atlas-kv-bmk-v17.8.0.9`
    - Automatically detects number of available CPUs
    - Output printed out, in evt/sec, with average, median, min, max
      - json output soon available (compatible with the benchmark suite format)

```
[root@bmk16-slc-e839aa4f-ca2b-4b3c-be68-dc6271caaf30 ~]# time docker run -it --rm gitlab-registry.cern.ch/giordano/hep-workloads:atlas-kv-bmk-v17.8.0.9
ncopies= 16
Using AtlasProduction/17.8.0.9 [cmt] with platform x86_64-slc6-gcc47-opt
        at /cvmfs/atlas.cern.ch/repo/sw/software/x86_64-slc6-gcc47-opt/17.8.0
**************************************************************************************************************************
KV cpu performance [evt/sec]: avg  0.758 over 16 threads. Median 0.758 Min Value 0.712 Max Value 0.793
**************************************************************************************************************************
```

    - Possibility to pin a subset of cores using Docker functionalities for more detailed studies,
      - `docker run -it --rm --cpuset-cpus=XXX gitlab-registry.cern.ch/giordano/hep-workloads:atlas-kv-bmk-v17.8.0.9`

  - NB: the duration is, as before, dominated by the athena initialization phase (6' vs ~2' of bmk events)
    - This time doesn't enter in the benchmark score
    - Could benefit from the work ongoing in ATLAS to snapshot the initialization phase

Just a proof of concept

- At least for the time being
  - Need feedback from you (site admins in particular) to validate the approach before continuing
    - Discuss a sustainable approach to distribute those candles
    - Verify that Docker approach meets expectations
  - Some advantages:
    - Easy to use
    - Possibility to run also with Container Orchestration Engines, and target large clusters
    - Can be integrated with monitoring tools to collect results

- Publicly available here, try it!
  - https://gitlab.cern.ch/giordano/hep-workloads/tree/master

To be done

- For ATLAS (already discussed with the “Performance understanding” team)
  - Need to adopt athena 21.x, and include reconstruction, or more complex sim events
- Try the same approach for the other experiments
  - Already started for CMS, based on the script used by Pepe Flix

- docker-volume-cvmfs also available: to be evaluated as a more sustainable alternative to snapshotting
  - https://gitlab.cern.ch/cloud-infrastructure/docker-volume-cvmfs
  - Allows also cvmfs tags usage, smaller Docker images, easier image build
  - Drawback: need to include a dry run to pre-fetch cvmfs libraries


CERN