Containers for ATLAS Benchmarking

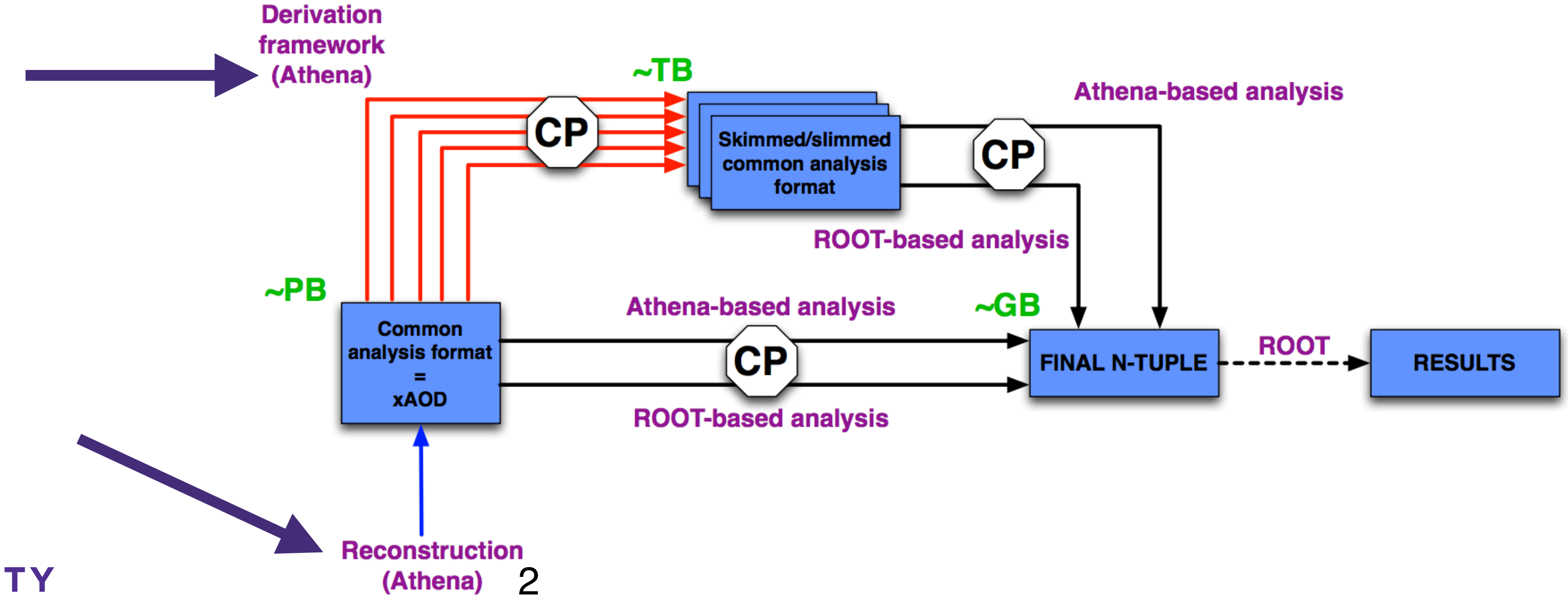
Lukas Heinrich

HEPiX meeting

NEW YORK UNIVERSITY

Atlas Computing Model

1) "Atlas Offline": Athena / Gaudi framework + associated packages
   - used for event generation, simulation and reconstruction.
     central production of configurable reduces dataset for analysis via "derivation framework"
2) Analysis Releases
   - used for physics analysis
   - currently two main frameworks a) standalone root-based b) athena-based



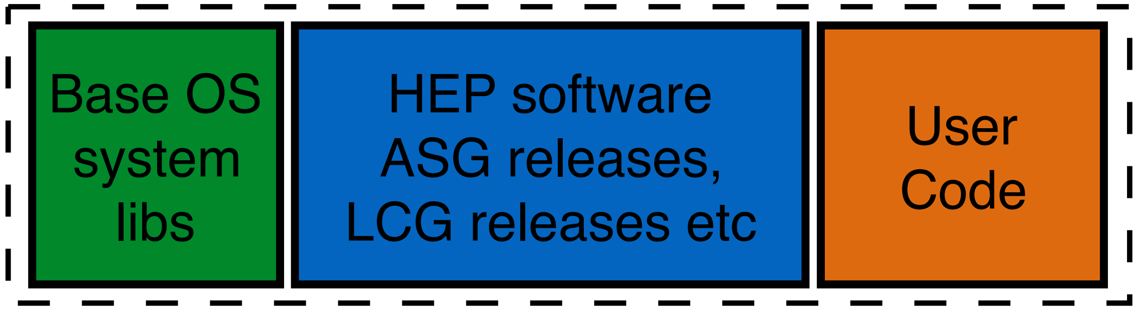
Container use-cases for ATLAS

- reproducible, interactive development environments for personal development / software tutorials
  - benchmarking (HEPiX) **new**
- development env / job env parity on distributed/batch systems
- continuous integration, release testing
- analysis preservation and reusability

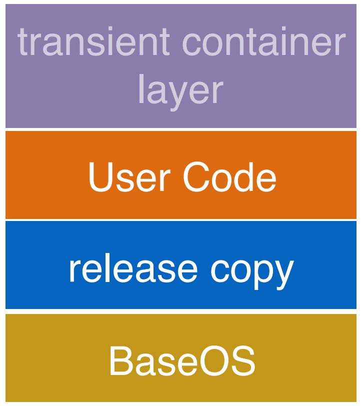
before we can use containers, we need images

Running ATLAS workloads in in Containers

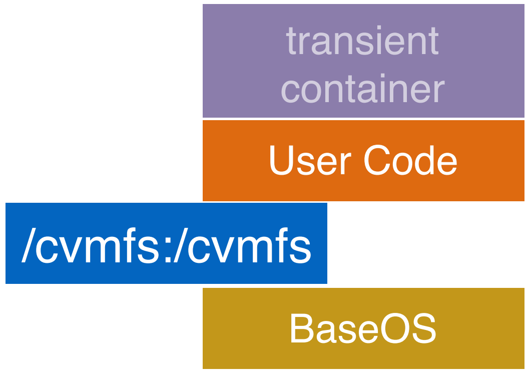
Three “macro” layers in s/w stack, various options how to get functioning container. Investigating all of them in ATLAS



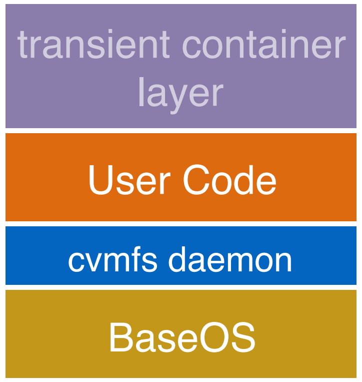
1) build full release into a container image



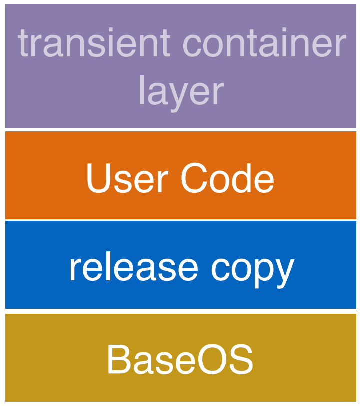
2) take release from external CVMFS mount (possibly tagged)



3) run cvmfs daemon inside container (won’t discuss)



Running ATLAS workloads in in Containers



1) build full release into a container image

Advantages:

best encapsulation, best for benchmarking since all (most) bits are in the image so do not suffer from network I/O caching issues for networked filesystem (cvmfs)

Disadvantages:

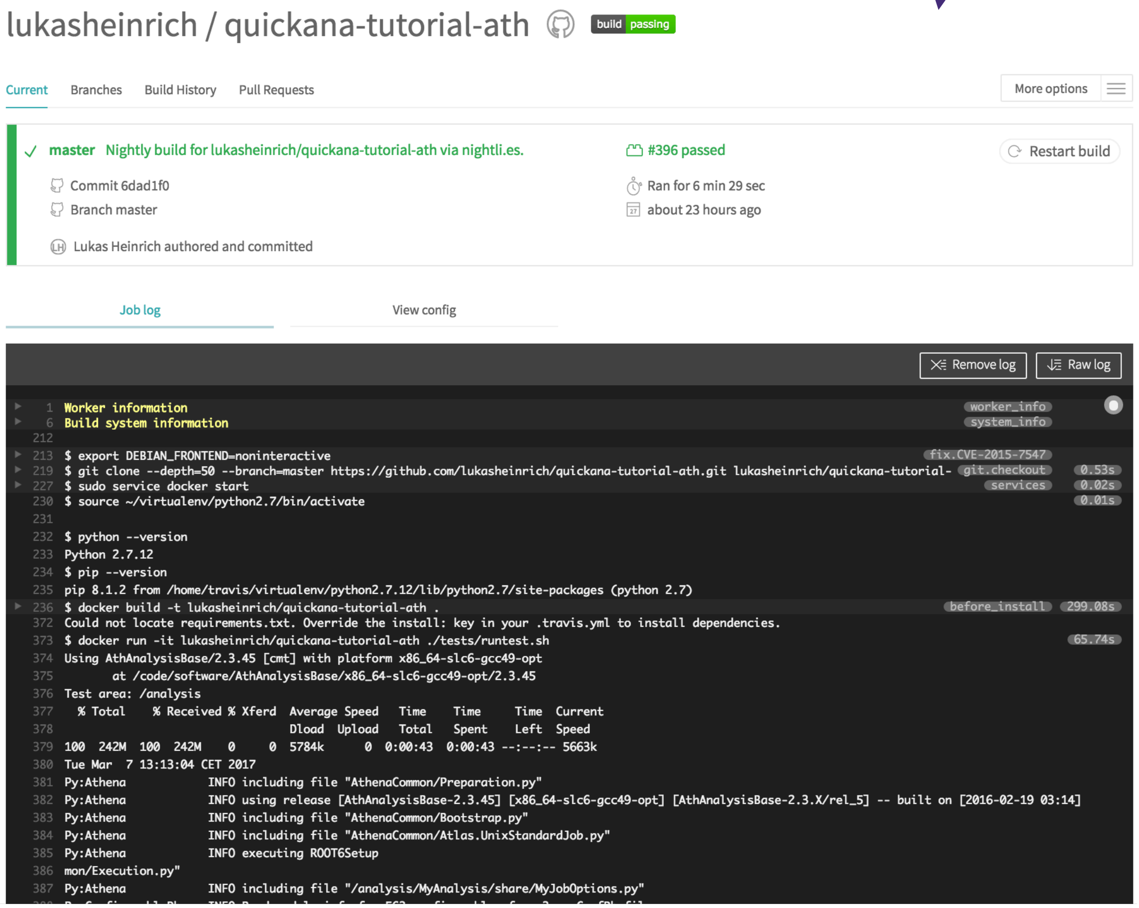
large image sizes 2-3GB for analysis releases, ~10GB for offline releases.

Experience:

built a number of analysis releases using cmt build system[1] (both flavors), works very well on top of official cern/cc7-base

started to investigate AtlasOffline installation of cmake releases with promising results

Athena running on Travis CI



✓ master Nightly build for lukasheinrich/quickana-tutorial-ath via nightli.es.

#396 passed

Commit 6dad1f0

Branch master

Ran for 6 min 29 sec

about 23 hours ago

Lukas Heinrich authored and committed

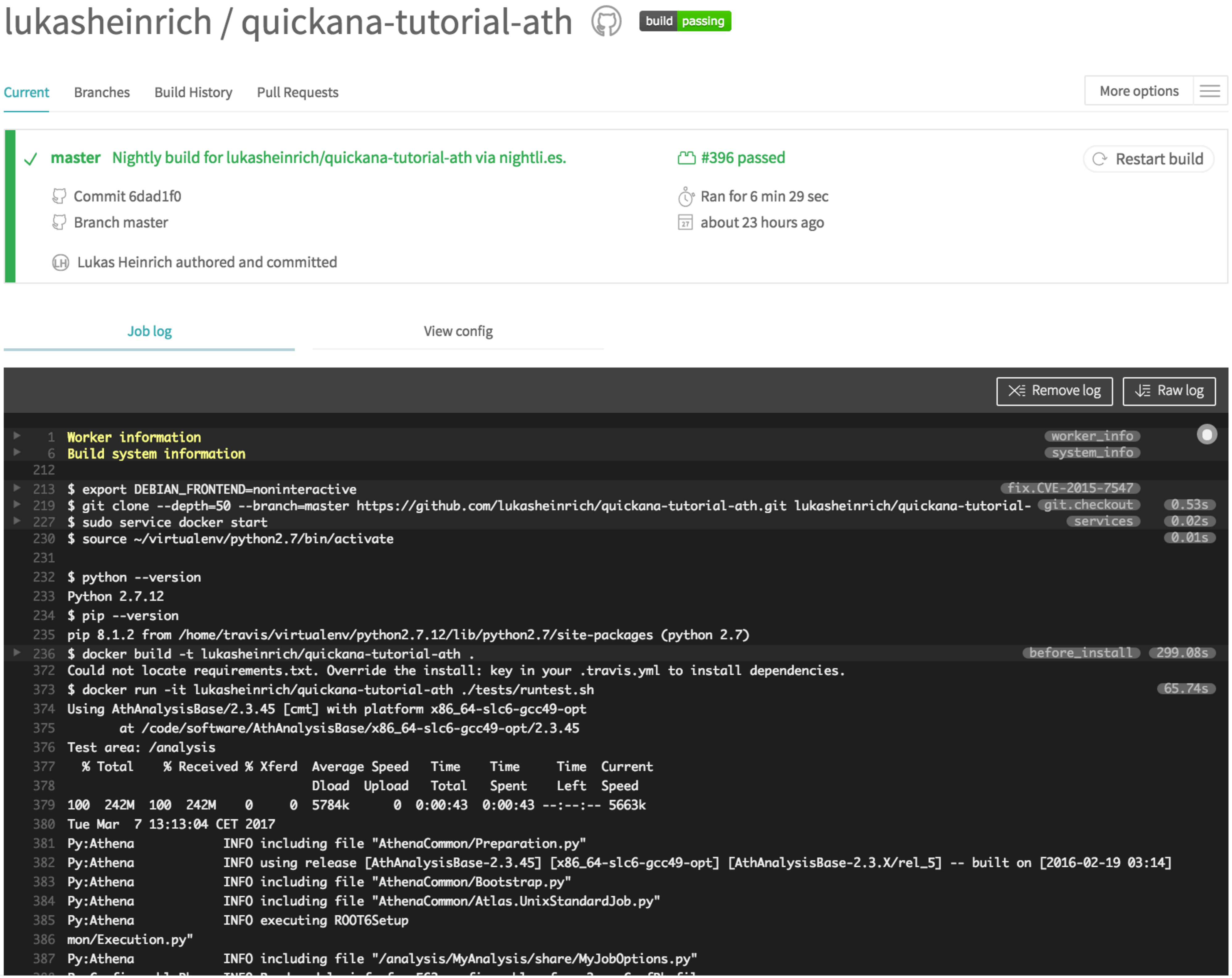
```
Worker information                                                                  worker_info
Build system information                                                            system_info

$ export DEBIAN_FRONTEND=noninteractive                                             fix.CVE-2015-7547
$ git clone --depth=50 --branch=master https://github.com/lukasheinrich/quickana-tutorial-ath.git lukasheinrich/quickana-tutorial-   git.checkout   0.53s
$ sudo service docker start                                                         services   0.02s
$ source ~/virtualenv/python2.7/bin/activate                                                   0.01s

$ python --version
Python 2.7.12
$ pip --version
pip 8.1.2 from /home/travis/virtualenv/python2.7.12/lib/python2.7/site-packages (python 2.7)
$ docker build -t lukasheinrich/quickana-tutorial-ath .                             before_install   299.08s
Could not locate requirements.txt. Override the install: key in your .travis.yml to install dependencies.
$ docker run -it lukasheinrich/quickana-tutorial-ath ./tests/runtest.sh                        65.74s
Using AthAnalysisBase/2.3.45 [cmt] with platform x86_64-slc6-gcc49-opt
       at /code/software/AthAnalysisBase/x86_64-slc6-gcc49-opt/2.3.45
Test area: /analysis
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  242M  100  242M    0     0  5784k      0  0:00:43  0:00:43 --:--:-- 5663k
Tue Mar  7 13:13:04 CET 2017
Py:Athena            INFO including file "AthenaCommon/Preparation.py"
Py:Athena            INFO using release [AthAnalysisBase-2.3.45] [x86_64-slc6-gcc49-opt] [AthAnalysisBase-2.3.X/rel_5] -- built on [2016-02-19 03:14]
Py:Athena            INFO including file "AthenaCommon/Bootstrap.py"
Py:Athena            INFO including file "AthenaCommon/Atlas.UnixStandardJob.py"
Py:Athena            INFO executing ROOT6Setup
mon/Execution.py"
Py:Athena            INFO including file "/analysis/MyAnalysis/share/MyJobOptions.py"
```

[1] Dockerfiles at github: lukasheinrich/asg-docker

Athena in ‘fat image’ (analsis release, not simulation/reconstruction)



lukasheinrich / quickana-tutorial-ath

✓ master Nightly build for lukasheinrich/quickana-tutorial-ath via nightli.es.

#396 passed

Commit 6dad1f0

Branch master

Ran for 6 min 29 sec

about 23 hours ago

Lukas Heinrich authored and committed

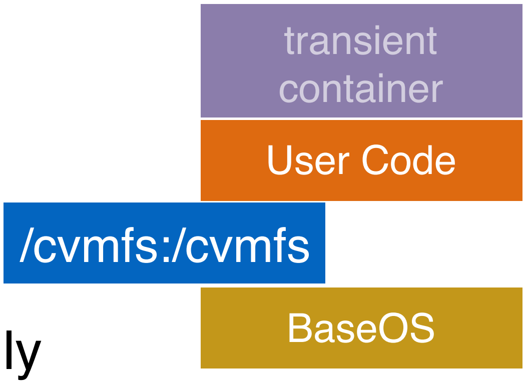
```
Worker information                                                                        worker_info
Build system information                                                                  system_info

$ export DEBIAN_FRONTEND=noninteractive                                             fix.CVE-2015-7547
$ git clone --depth=50 --branch=master https://github.com/lukasheinrich/quickana-tutorial-ath.git lukasheinrich/quickana-tutorial-  git.checkout   0.53s
$ sudo service docker start                                                                  services   0.02s
$ source ~/virtualenv/python2.7/bin/activate                                                            0.01s

$ python --version
Python 2.7.12
$ pip --version
pip 8.1.2 from /home/travis/virtualenv/python2.7.12/lib/python2.7/site-packages (python 2.7)
$ docker build -t lukasheinrich/quickana-tutorial-ath .                             before_install   299.08s
Could not locate requirements.txt. Override the install: key in your .travis.yml to install dependencies.
$ docker run -it lukasheinrich/quickana-tutorial-ath ./tests/runtest.sh                                 65.74s
Using AthAnalysisBase/2.3.45 [cmt] with platform x86_64-slc6-gcc49-opt
       at /code/software/AthAnalysisBase/x86_64-slc6-gcc49-opt/2.3.45
Test area: /analysis
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  242M  100  242M    0     0  5784k      0  0:00:43  0:00:43 --:--:-- 5663k
Tue Mar  7 13:13:04 CET 2017
Py:Athena            INFO including file "AthenaCommon/Preparation.py"
Py:Athena            INFO using release [AthAnalysisBase-2.3.45] [x86_64-slc6-gcc49-opt] [AthAnalysisBase-2.3.X/rel_5] -- built on [2016-02-19 03:14]
Py:Athena            INFO including file "AthenaCommon/Bootstrap.py"
Py:Athena            INFO including file "AthenaCommon/Atlas.UnixStandardJob.py"
Py:Athena            INFO executing ROOT6Setup
mon/Execution.py"
Py:Athena            INFO including file "/analysis/MyAnalysis/share/MyJobOptions.py"
```

[1] Dockerfiles at github: lukasheinrich/asg-docker

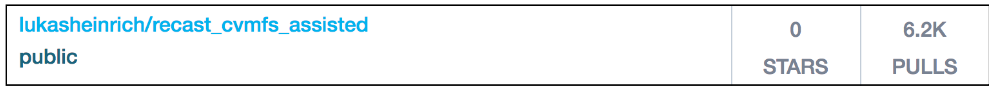
Running ATLAS workloads in in Containers



2) cvmfs from external mount

Advantages:
easily achievable, just need base layer → small image sizes (even with user code).
No need to additional image management, can piggy-back off of cvmfs installations, only occasional updates to base image.

Disadvantages:
breaks encapsulation. container unusable without cvmfs. Difficult to add User Code layer (cannot use Dockerfile if you want to to non-trivial things, like compile user code using cvmfs. Workarounds by committing running containers.



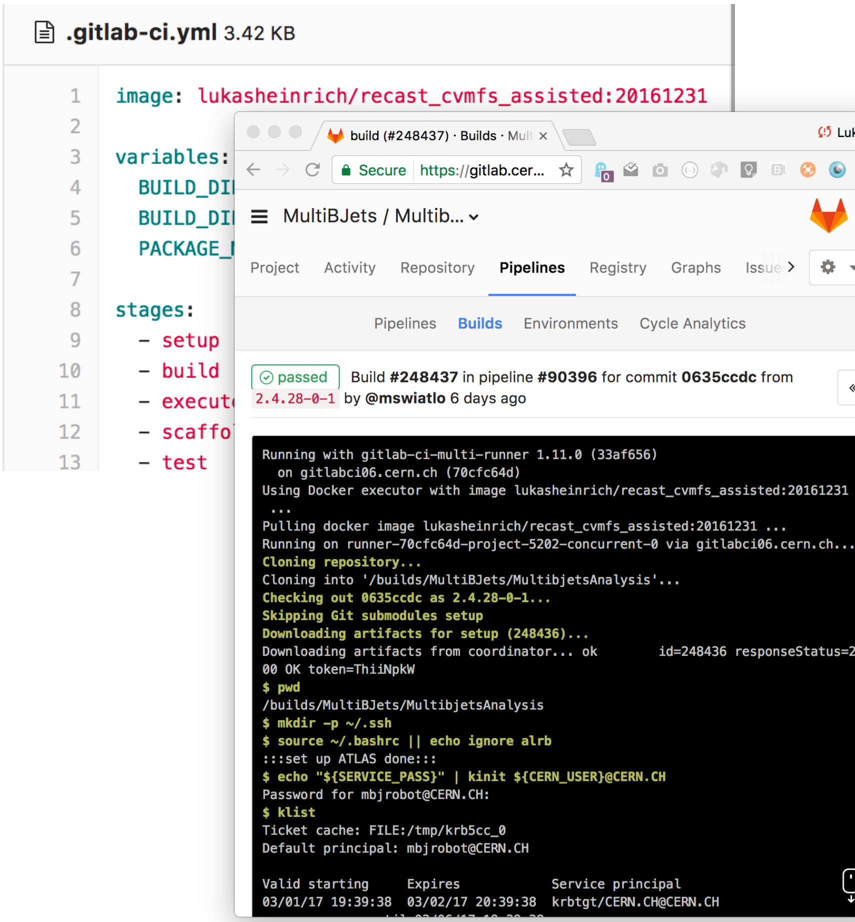
lukasheinrich/recast_cvmfs_assisted
public

0 STARS
6.2K PULLS

Ops Considerations:
Very good experience with CVMFS volume plugin
Used for example by GitLab to provide CVMFS. Mounting /cvmfs directly tricky when auto-mounting (stale file descriptors)

encapsulation breakage can be mitigated by using specific cvmfs tag

```
-v atlas.cern.ch#<some-hash>:/cvmfs/atlas.cern.ch
-v atlas.cern.ch@trunk-previous:/cvmfs/atlas.cern.ch
```



.gitlab-ci.yml 3.42 KB

```
image: lukasheinrich/recast_cvmfs_assisted:20161231

variables:
  BUILD_DI
  BUILD_DI
  PACKAGE_

stages:
  - setup
  - build
  - execut
  - scaffo
  - test
```

passed Build #248437 in pipeline #90396 for commit 0635ccdc from 2.4.28-0-1 by @mswiatlo 6 days ago

```
Running with gitlab-ci-multi-runner 1.11.0 (33af656)
  on gitlabci06.cern.ch (70cfc64d)
Using Docker executor with image lukasheinrich/recast_cvmfs_assisted:20161231
 ...
Pulling docker image lukasheinrich/recast_cvmfs_assisted:20161231 ...
Running on runner-70cfc64d-project-5202-concurrent-0 via gitlabci06.cern.ch...
Cloning repository...
Cloning into '/builds/MultiBJets/MultibjetsAnalysis'...
Checking out 0635ccdc as 2.4.28-0-1...
Skipping Git submodules setup
Downloading artifacts for setup (248436)...
Downloading artifacts from coordinator... ok        id=248436 responseStatus=2
00 OK token=ThiiNpkW
$ pwd
/builds/MultiBJets/MultibjetsAnalysis
$ mkdir -p ~/.ssh
$ source ~/.bashrc || echo ignore alrb
:::set up ATLAS done:::
$ echo "${SERVICE_PASS}" | kinit ${CERN_USER}@CERN.CH
Password for mbjrobot@CERN.CH:
$ klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: mbjrobot@CERN.CH

Valid starting     Expires            Service principal
03/01/17 19:39:38  03/02/17 20:39:38  krbtgt/CERN.CH@CERN.CH
```

Running standard ATLAS simulation using docker

```
docker run --rm -it --security-opt label:disable  \
  -v atlas.cern.ch:/cvmfs/atlas.cern.ch \
  -v atlas-condb.cern.ch:/cvmfs/atlas-condb.cern.ch \
  -v sft.cern.ch:/cvmfs/sft.cern.ch \
  -v volone:/data \
  -w /data
  lukasheinrich/athena_trfsbase bash
```

```
$> source /code/resource/env.sh 20.3.7.4
$> Sim_tf.py '--inputEVNTFile' '/data/mu_E50_eta0-25.evgen.pool.root' \
             '--outputHITSFile' 'HITSMPSim.pool.root' '--maxEvents' '4' \
             '--skipEvents' '0' '--randomSeed' '10' '--geometryVersion' 'ATLAS-R2-2015-03-01-00_VALIDATION' \
             '--conditionsTag' 'OFLCOND-RUN12-SDR-19' '--DataRunNumber' '222525' '--physicsList' 'FTFP_BERT' \
             '--postInclude' 'AtlasG4Tf:G4AtlasTests/postInclude.DCubeTest.py' \
             '--preExec' 'AtlasG4Tf:simFlags.ReleaseGeoModel=False' '--simulator=MC12G4' '--useISF=True'
```

Running standard ATLAS simulation using docker

basic benchmarks already written out as part of ‘jobReport.json’ in worker

```
cat jobReport.json|jq '.resource.transform'
{
  "cpuEfficiency": 0.2715,
  "cpuPWEfficiency": 0.2715,
  "cpuTime": 3,
  "cpuTimeTotal": 80,
  "externalCpuTime": 1,
  "processedEvents": 4,
  "wallTime": 302
}
```

```
cat jobReport.json|jq '.resource.executor.EVNTtoHITS'
{
  "cpuTime": 79,
  "dbData": 4211342,
  "dbTime": 57.88,
  "memory": {
    "Avg": {
      "avgPSS": 542864,
      "avgRSS": 547724,
      "avgSwap": 0,
      "avgVMEM": 1010683
    },
    "Max": {
      "maxPSS": 956955,
      "maxRSS": 962336,
      "maxSwap": 0,
      "maxVMEM": 1580940
    }
  },
  "nevents": 4,
  "postExe": {
    "cpuTime": 0,
    "wallTime": 1
  },
  "preExe": {
    "cpuTime": 0,
    "wallTime": 0
  },
  "total": {
    "cpuTime": 80,
    "wallTime": 297
  },
  "validation": {
    "cpuTime": 0,
    "wallTime": 1
  },
  "wallTime": 296
}
```

Running standard ATLAS simulation using docker

can also run ‘standard transform’ configurations based on configuration database (AMI — Atlas Metadata Interface), but needs ATLAS VOMS authentication

VOMS proxy can be mounted into the container without a problem

```
$> Sim_tf.py --AMI sXXXX \
   --inputEVNTFile /data/mu_E50_eta0-25.evgen.pool.root \
   --outputHITSFile HITSMPSim.pool.root --maxEvents 4
```

Successfully tested running ATLAS full chain MC generation → simulation → reconstruction → derivation completely in containers.

Container Runtime considerations

on-going discussion on choice between container run-time. e.g. Docker vs Singularity vs Shifter. Goal is to build OCI (Open Container Initiative)-compatible images and convert to custom non-OCI runtime formats when necessary

Singularity and Shifter used as runtimes in HPC community

- Tested running self-contained athena installation via Shifter. Needed custom conversion, but automatized via `shifterimg pull`
- Singularity can read from Docker Hub directly as well

Runtime switch expected to work transparently / be an implementation detail. Focus for ATLAS is on providing images

NB: Singularity may appear soon on LXPLUS

Roadmap

ATLAS plans to build fully self-contained installation of ATLAS software releases as docker images with the next major releases.

This should enable benchmarking on generic computing infrastructure that can run containers without further setup without the need for e.g. Kit Installation

In the meantime, benchmarking by using container + cvmfs volume plugin is possible, but may need cache warm-up

Appendix