ACAT 2017

Using Containers with ATLAS Offline Software

Marcelo Vogel
on behalf of the ATLAS Collaboration

BERGISCHE UNIVERSITÄT WUPPERTAL

ATLAS EXPERIMENT

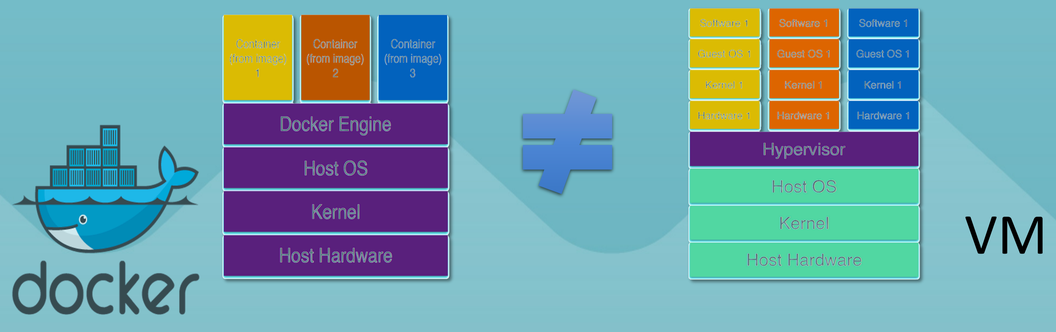
Introduction to Docker Containers

Why Docker? Efficiently ship & quickly launch applications on cloud computing resources

What is Docker? Software-based environment and resource isolation technology

How does it work? Processes run natively (but isolated) on minimally configured hosts

Docker is not a virtual Machine, which simulates both hardware and software. Docker will share the OS kernel, but isolate different environments ("containers") with respect to data, code, networking, etc.

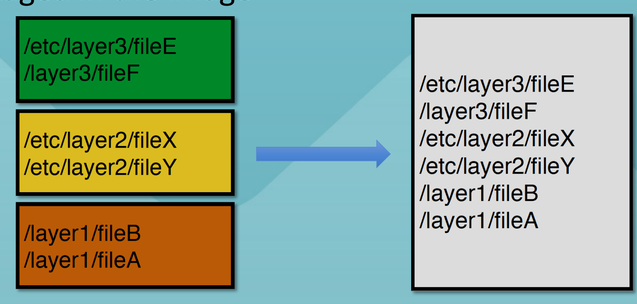


Docker images: archived, executable file system snapshots

Images are portable, self-contained, reproducible software environments. An image includes full dependencies (OS, libraries, user code)

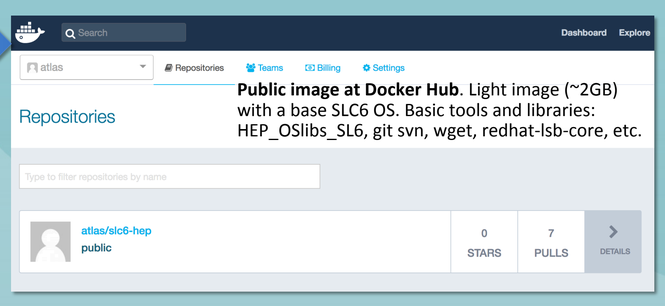
Docker images are built from layers

A container is an instance of an image, which starts as a single process in the environment packaged in the image



Dockerfile: Automatic Image Building

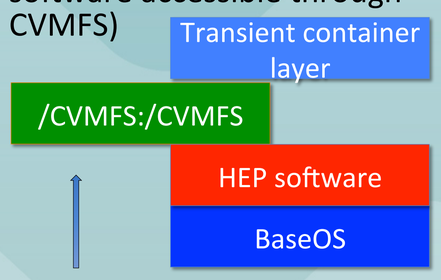
Images can be built automatically by Docker if provided with a **Dockerfile**, which is basically a set of instructions to be executed in command-line fashion (Maintained at gitlab.cern.ch)



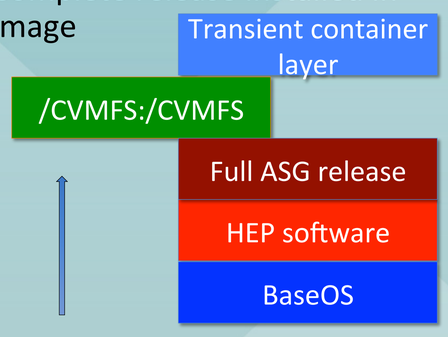
Public image at Docker Hub. Light image (~2GB) with a base SLC6 OS. Basic tools and libraries: HEP_OSlibs_SL6, git svn, wget, redhat-lsb-core, etc.

We also build standalone images!

Light image (~2GB). ATLAS software accessible through CVMFS)



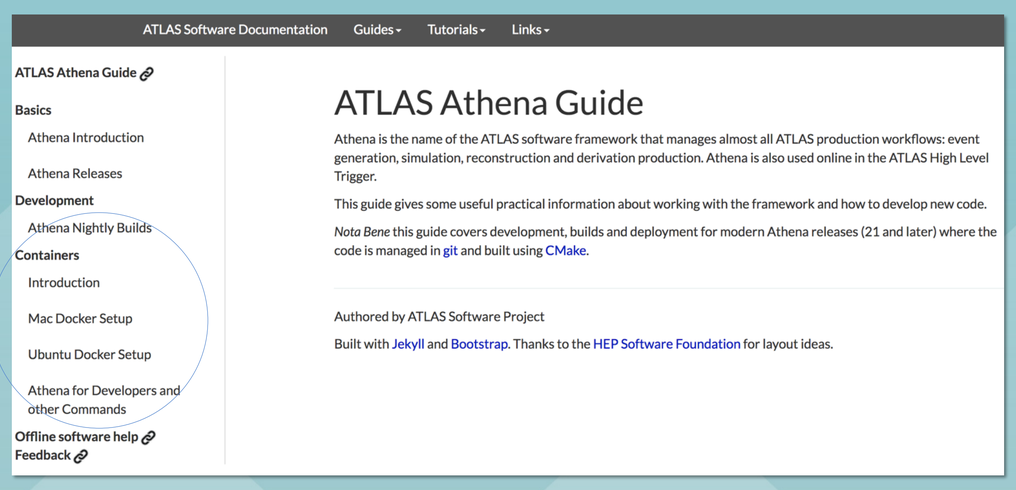
Full release image (~22GB) Complete release installed in image



CVMFS is bind mounted to unprivileged containers via the CVMFS Docker Volume plugin supported by CERN IT. CVMFS is installed in the host system and the cache can be shared across containers

ATLAS Software in Docker Images

The new repository for ATLAS computing documentation has a new section for working with Docker containers



The guide provides detailed instructions for building, running and managing containers with ATLAS software. For example, you can clone the ATLAS Docker repository

git clone https://:@gitlab.cern.ch:8443/atlas-sit/docker.git

build an image and clone the Git Athena repository inside the container. Then you are ready to develop software!

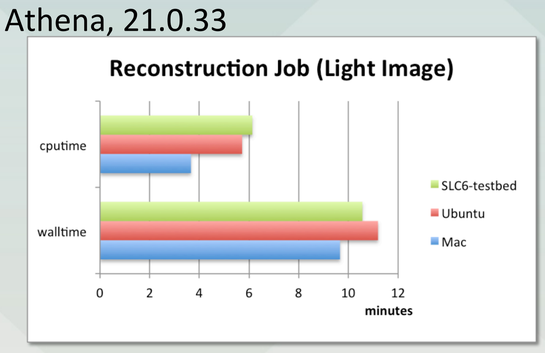
git clone https://[YOUR_USER_NAME]@gitlab.cern.ch/[YOUR_USER_NAME]/athena.git

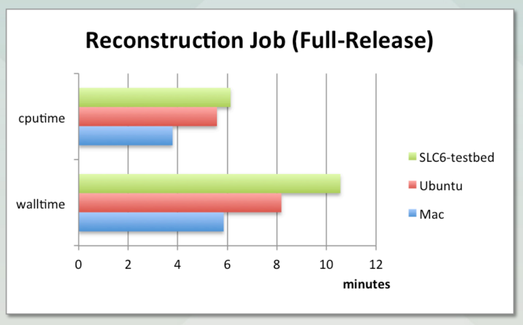
Performance of Containers Running ATLAS Software

Simple tests comparing Wall-time and CPU-time in reconstruction jobs using the light and full-release images. Docker is tested in two platforms frequently used by developers: **macOS and Ubuntu**

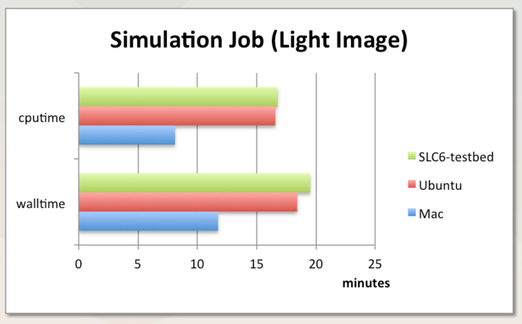
Latest version of Docker for Mac, version 17.06.0 (stable). Docker version 17.05.0 in Ubuntu. **Ubuntu 14.04** desktop, Intel Core 2 DUO CPU E7500, 2.93GHz (CVMFS 2.3.5). **macOS Sierra 10.12.6**, MacBook Pro (Retina, 15-inch, Mid 2014), i7 2.2 GHz (CVMFS 2.3.3, FUSE 3.5.4)

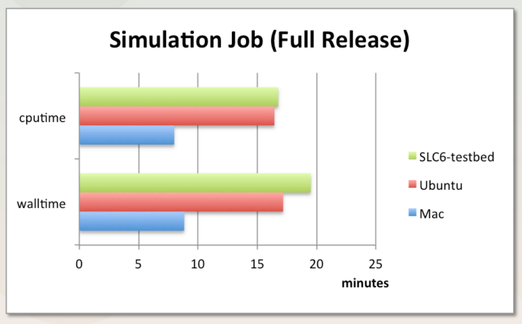
Reconstruction test job (q431, RAW to ESD) on 25 data events (data16_13TeV, physics_Main) in Athena, 21.0.33





Full-simulation test job (EVNT to HITS, 5 events)



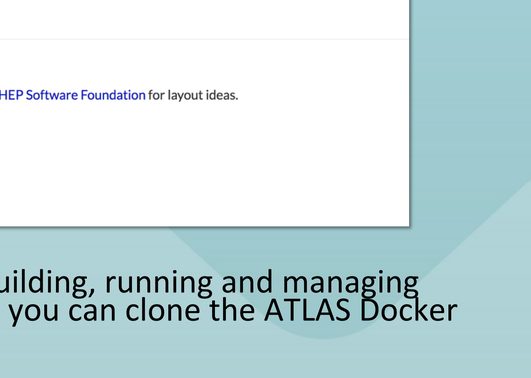


Consistency in Bind Mounts

Consistent: perfect consistency. The host and the container have an identical view of the mount at all times

Cached: the host view is authoritative. Delays are allowed before updates in the host appear in the container

Delegated: the container view is authoritative. Delays are allowed before updates in the container appear in the host



"Benchmark" script: testMergeCont.C, a tutorial macro from ROOT 6.08.02, is used to evaluate the "speed" of different setups. It creates random histograms on a canvas, stores them in a ROOT file and reads them back