Common Software Platform CernVM/-FS Perspective

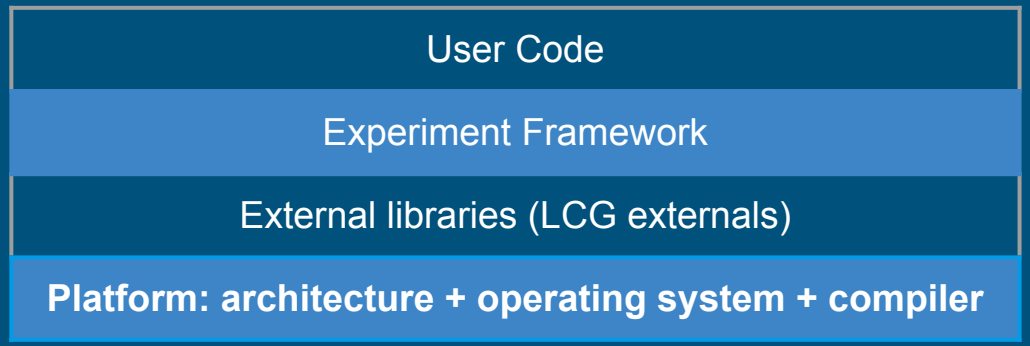
(Informed by Software Deployment pre-GDB: https://indico.cern.ch/event/813800)

AF, 9 July 2020

Jakob Blomer

Software stack

- CernVM-FS:
  efficient distribution of
  platform, externals, framework
- Facilitates **proliferation** of
  combinations



- CernVM: aims at **consolidating** the platform to a single, common OS layer
  - Full VM virtualization (graphical & batch)
  - Runtime container (not derivable, requires cvmfs)
  - Defined by the experiment application

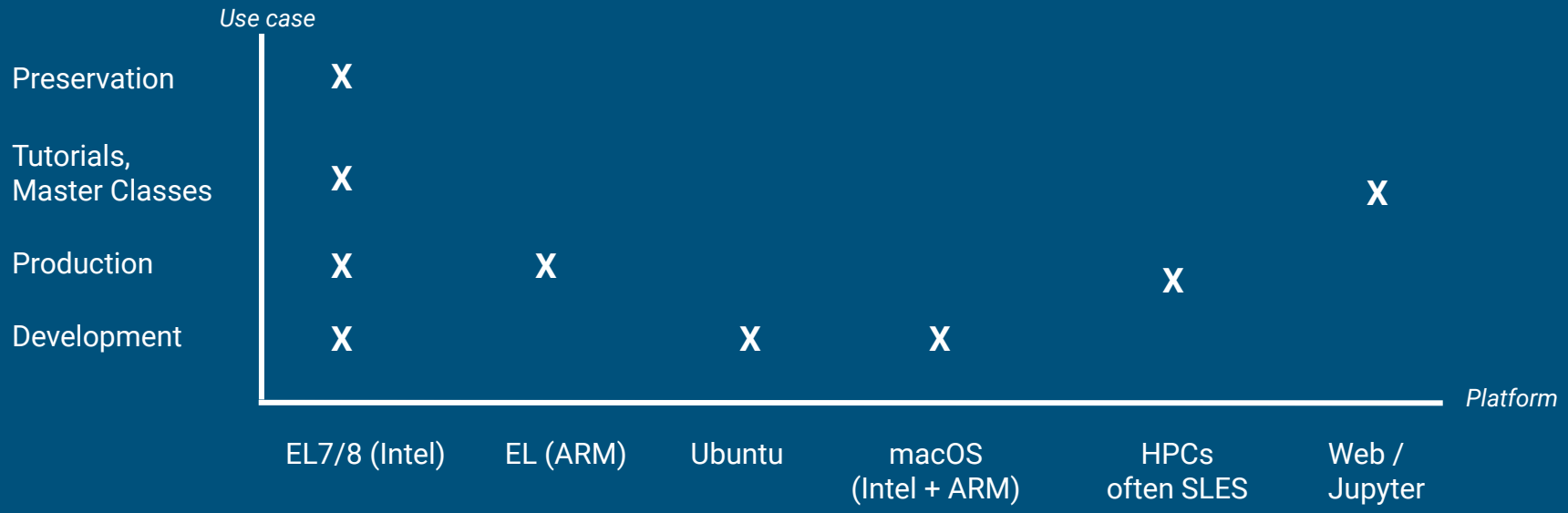
Reminder: pre-GDB summary I/II

- We should expect the future grid worker node to be Linux + cvmfs only
- Software comes as a container from cvmfs
  - Currently singularity is the container runtime of choice but that might change in the future
- Worker nodes do not have all the same hardware
  - Different micro-architecture and accelerator options
  - Best software stack for given architecture selected at runtime: approaches for runtime selection of binaries from LCG, CMS, and the Spack package manager
- HEP_OSlibs is moving from a runtime dependency to a potential build dependency for container creators
  - HEP_OSlibs formalizes the boundary between operating system and application software
  - In general: the boundary between OS and application software is currently up to negotiation, we should investigate if this process can be automated

Reminder: pre-GDB summary II/II

- Containers dominate the resource space, full VMs used more and more in niches (e.g. some T2s, ATLAS HLT, some outreach projects)
- Several different models to create containers and software stacks
  - Librarians rely for some containers on the LCG stack, sometimes on HEP_OSlibs
  - Spack becomes a popular tool for librarians: LCG, Key4HEP, LHCb, ALICE looking into it
  - User containers typically built freely off a base image, some concerns about image proliferation
  - Current stand-alone containers might get smaller in size with better layer management
- HEP wide container registry with optimal cvmfs integration seen as useful
  - CERN registry currently bound to internal gitlab projects
  - Dockerhub, being commercial, not ideal for large-scale HEP use cases

Platform dimensions



Several platforms serve a relatively narrow use case,
yet every platform comes with a support effort

Virtualization

While created to streamline OS combinatorics, virtualization technology has added its own technology options on top

Full virtualization (VM)

- Maximum freedom because we control kernel: cvmfs, eos built-in
- Difficult to create and maintain
- Graphics support
- Performance penalty

- Infrastructure streamlined over time to a few de-facto standard hypervisors: VirtualBox, OpenStack, EC2, maybe Azure (last three very similar)

Containers

- Easy to create, also by regular users
- Easy to create a mess of many, “half-baked” containers
- Graphics tricky (X11 forwarding etc.)
- No performance penalty (on Linux)

- Infrastructure still settling: **singularity**, **docker**, podman, **k8s**, containerd, runc, …
- Each container runtime requires a little extra thought for cvmfs support
- New Linux kernels should allow in-container cvmfs mount

Questions for SFT and Experiments

We are quite successfully exploring different platforms.
Can we **converge** to a (**common**) direction for platforms and virtualization technologies?

1. To which extent do we still need full virtualization
   - Seems still beneficial for graphics / tutorials and some commercial clouds
     - Are future tutorials fully web-based? Future commercial clouds k8s only?
   - Still important for long-term preservation, simple containers will quickly be unable to talk to the infrastructure (HTTPS, XRootD etc.)
   - Suggestion for a future CernVM: minimal, graphical VM that runs applications in containers from cvmfs (requires some investment!)
2. Can we reduce the number of supported OSs
   - Focus on containerized environments for development machines (with editor on the host)
3. Is there a well-defined step to “normalize” user containers after exploration phase
   - Physics results based on random user containers are impossible to preserve in the longer term (we can only “capture the chaos”)
4. Do we want to invest in a common platform or do experiments rather want to control the platform themselves