CernVM-FS powered container hub

Enrico Bocchi

Jakob Blomer

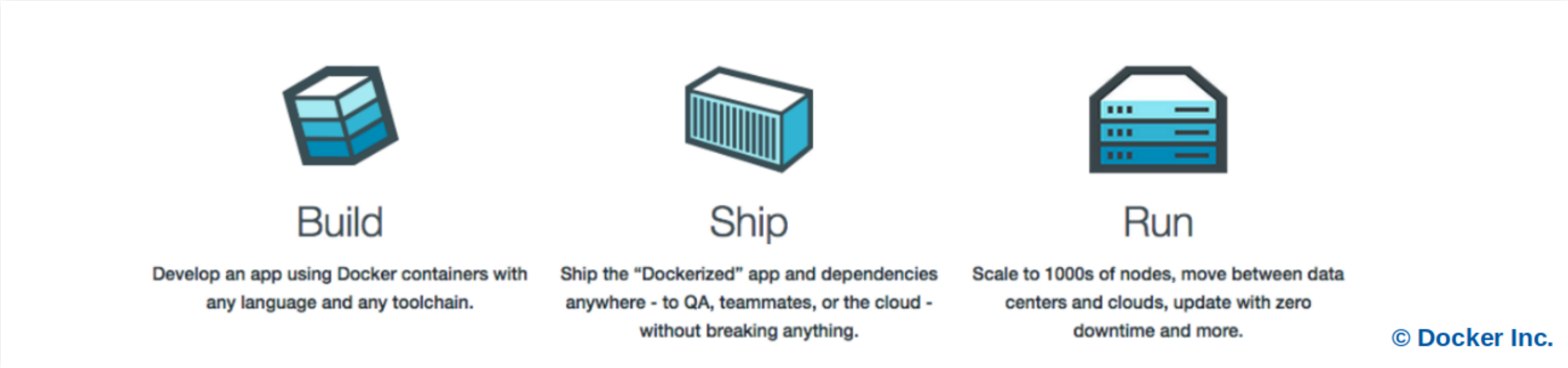
Simone Mosciatti

Andrea Valenzuela

vCHEP 2021, 19th May

The containers ecosystem

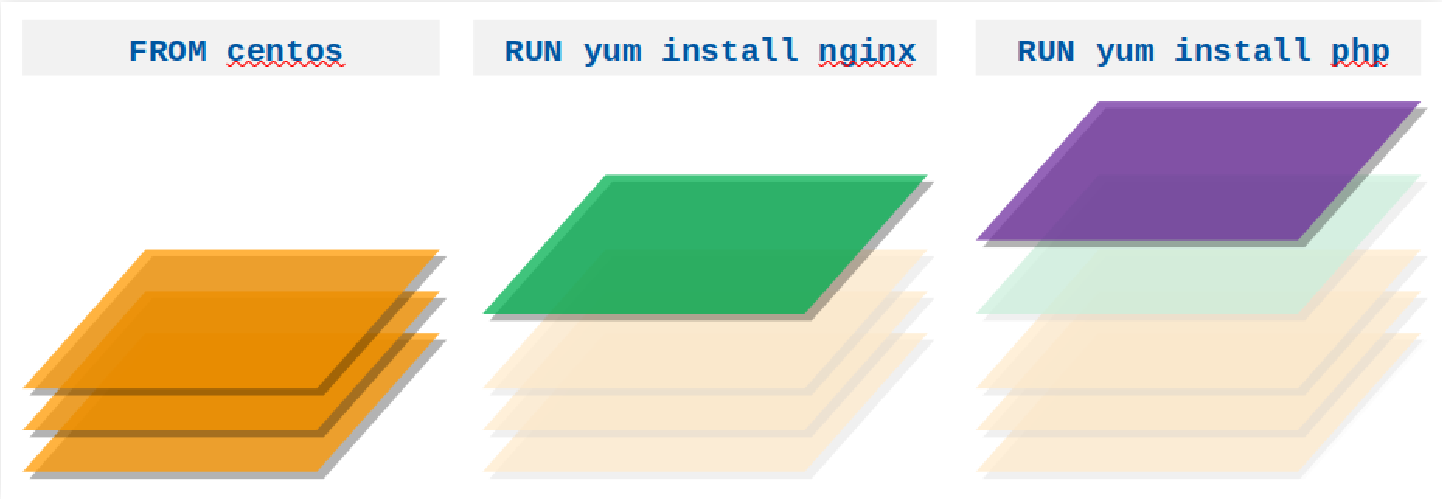
1. **Images**:
    - Immutable units with binaries, dependencies, ...
2. **Registries**:
    - Specialized repositories where to store images
3. **Runtimes**:
    - Software required to run container images



Containers in HEP

1. **Reproductibility**
    - Images freeze software and tools
    - Re-run the container to reproduce the analysis
2. **Portability and execution at scale**
    - Run containers on heterogeneous resources
    - Take advantage of computational power on WLCG
3. **Facilitate exploratory analysis**
    - Scientists encapsulate analysis code in containers
    - Validate on a single machine, then distribute at scale

Container images are a collection of layers




```
# docker history myimage
IMAGE           CREATED         CREATED BY                               SIZE
75cc2375258a    4 seconds ago   /bin/sh -c yum -y install php            66.9MB
e779b8a4024f    9 seconds ago   /bin/sh -c yum -y install nginx          77.8MB
470671670cac    4 days ago      /bin/sh -c #(nop)  CMD ["/bin/bash"]     0B
<missing>       4 days ago      /bin/sh -c #(nop)  LABEL org.label…      0B
<missing>       7 days ago      /bin/sh -c #(nop) ADD file:aa54047…      237MB
```

Limitations in images distribution

1. **Layers reduce deduplication efficiency**
   - Deduplication with coarse, per-layer granularity
2. **Network overhead to transfer images**
   - Big images increase network load
   - Longer waiting time to run the container
3. **Images are cached on local disk**
   - Runtimes require access to images to run them
   - Local storage can be scarce (e.g., HPC environments)
   - Storage space is not reclaimed when image is unused

CVMFS for global software distribution

Container images support in CVMFS

- ***DUCC***: Server-side component to ingest existing images
  - Unpacks images into flat filesystem
  - Applies file-based deduplication and hashing
  - Creates directory structure and publishes
- Regulation of ingestion via:
  - Wishlist
  - Webhook notification
    - Traditional registries notify CVMFS via HTTP
    - Integration demonstrated with Harbor

CVMFS integration with container runtimes

Runtime	**Type**	**CVMFS Support**
Singularity	Flat (+Layers)	Native
Docker	Layers	via *Graph Driver* plugin[1]
containerd[2]	Layers	via *Snapshotter* plugin
Podman	Layers	via *Additional image stores*[3]

[1] *Graph Driver* to converge on *Snapshotter* plugin soon

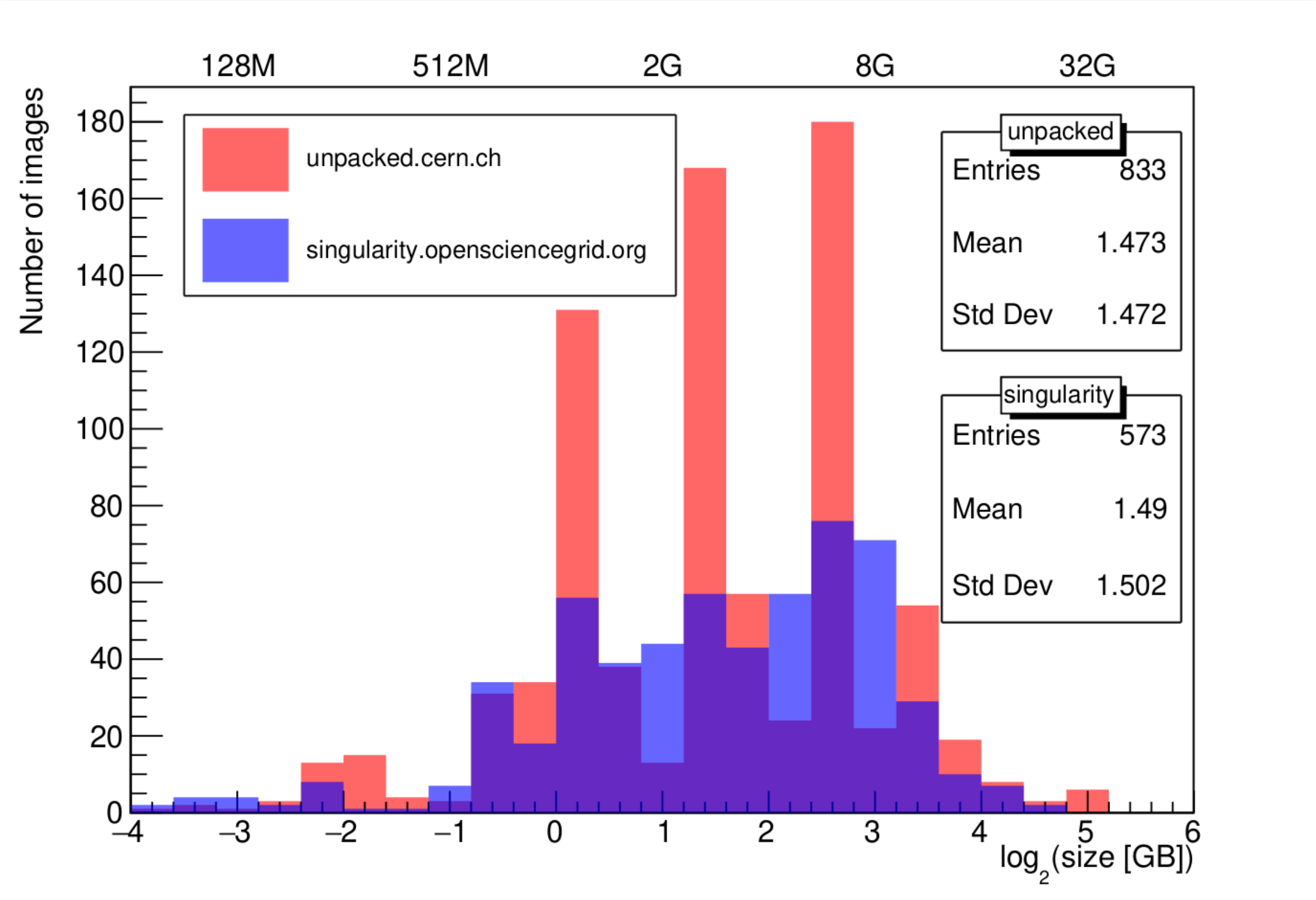
[2] containerd is supported by Kuberntes

[3] https://www.redhat.com/sysadmin/image-stores-podman

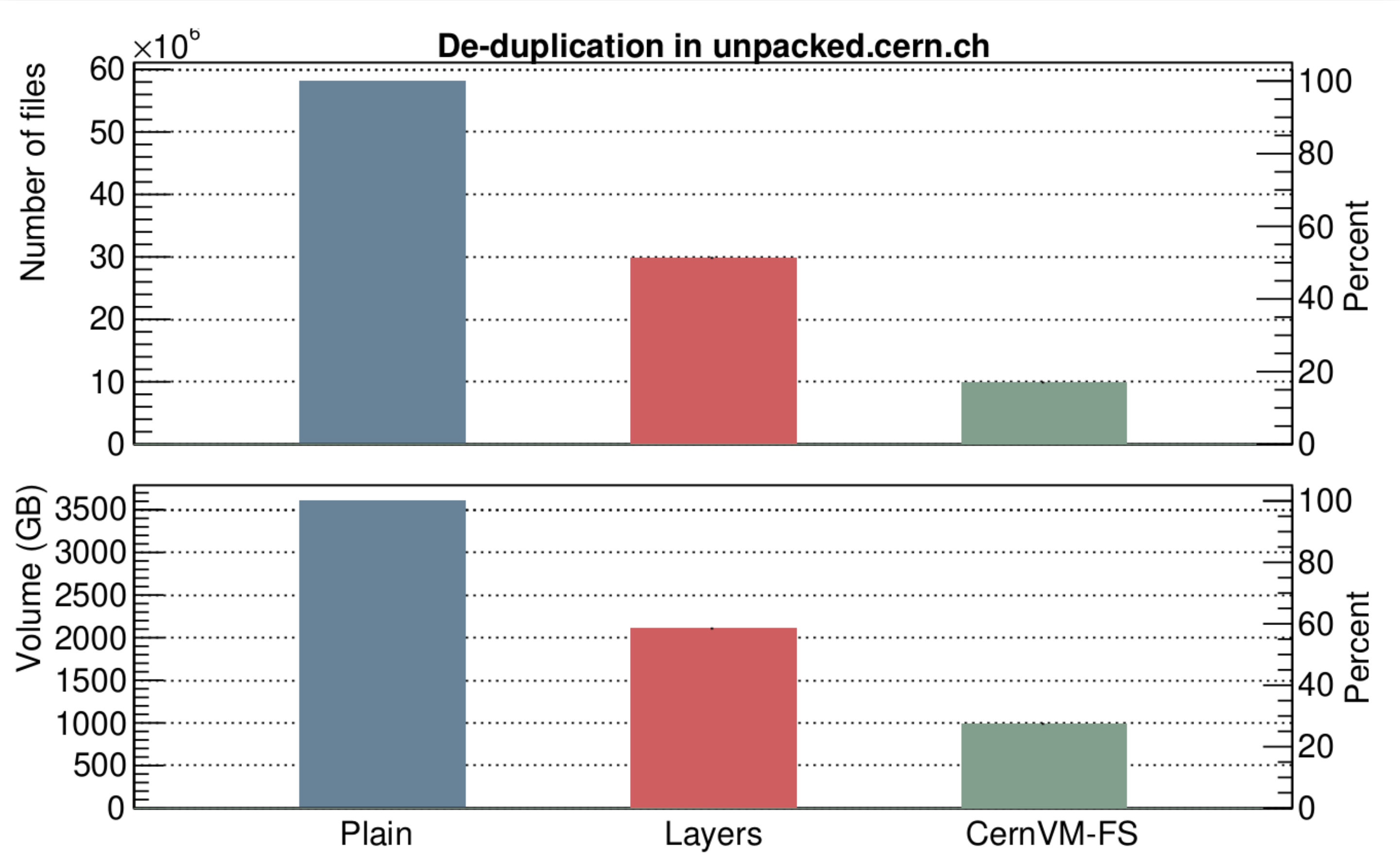
CVMFS-powered container registries

- `cvmfs/unpacked.cern.ch`
  - 800+ images
  - 3.5 TB, 50 M files
- `cvmfs/singularity.opensciencegrid.org`
  - 500+ images

Distribution of image sizes



Comparison of deduplication efficiency



Folding@Home

- Example of large-scale deployment
- Runs on the grid off containers served from `/cvmfs`

Conclusions

- Containers are mainstream technology
- Widely used by scientists in HEP
    - Reproducibility
    - Portability on heterogeneous resources
    - Exploratory analysis

Conclusions

- CVMFS is fully compatible with existing resources
  - Ingest and distribute available images
  - Support multiple container runtimes
  - Maintain isolation properties of standard containers
- … and hugely improves on storage and distribution efficiency
  - More efficient file-based deduplication
  - CVMFS clients cache only required content on-demand
  - Re-use existing CDN and on-site expertise

Backup

Containers in HEP

1. **Base Images**
    - Contain the bare operating system
    - Small in size, change rarely
2. **Experiment Images**
    - Contain the software stack of an experiment
    - Big (many dependencies) and updated weekly
3. **User Images**
    - Perform one specific task or analysis on data
    - Bigger and subject to frequent changes
        - Multiple times a day during development

Regulating ingestion of images

1. **Wishlist**
   - Users express interest in images to be ingested
   - DUCC verifies if the repository content is up-to-date
2. **Webhook notification**
   - Traditional registries notify CVMFS via HTTP
   - DUCC intercepts the webhook and starts the ingestion
   - ✔ Automates publication on registries and CVMFS
   - ✔ Integration demonstrated with Harbor