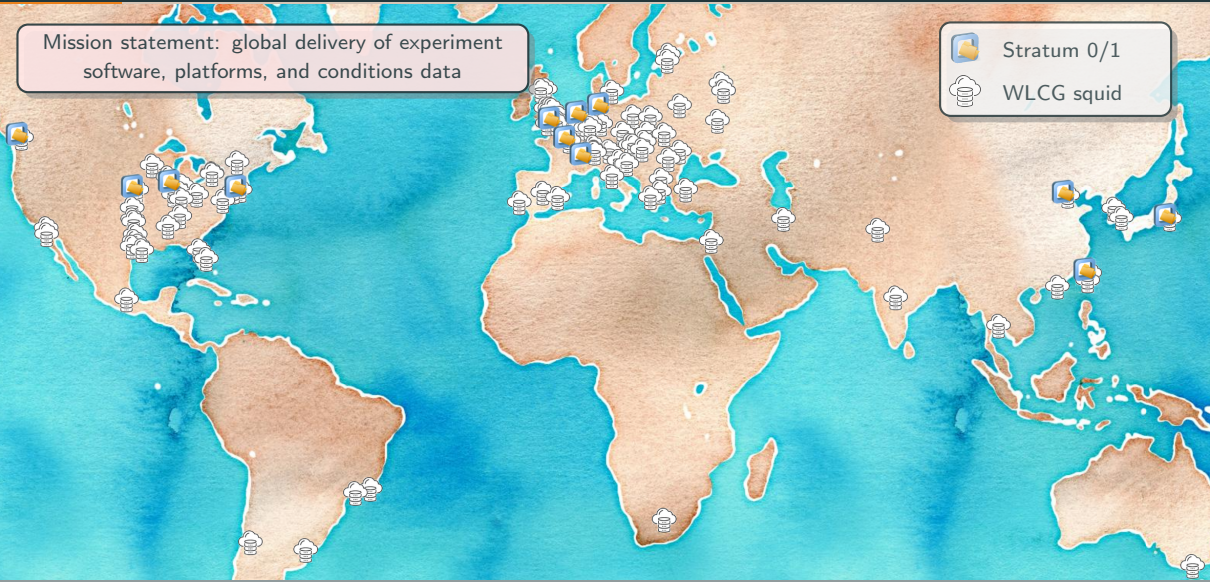
CernVM-FS for Containers

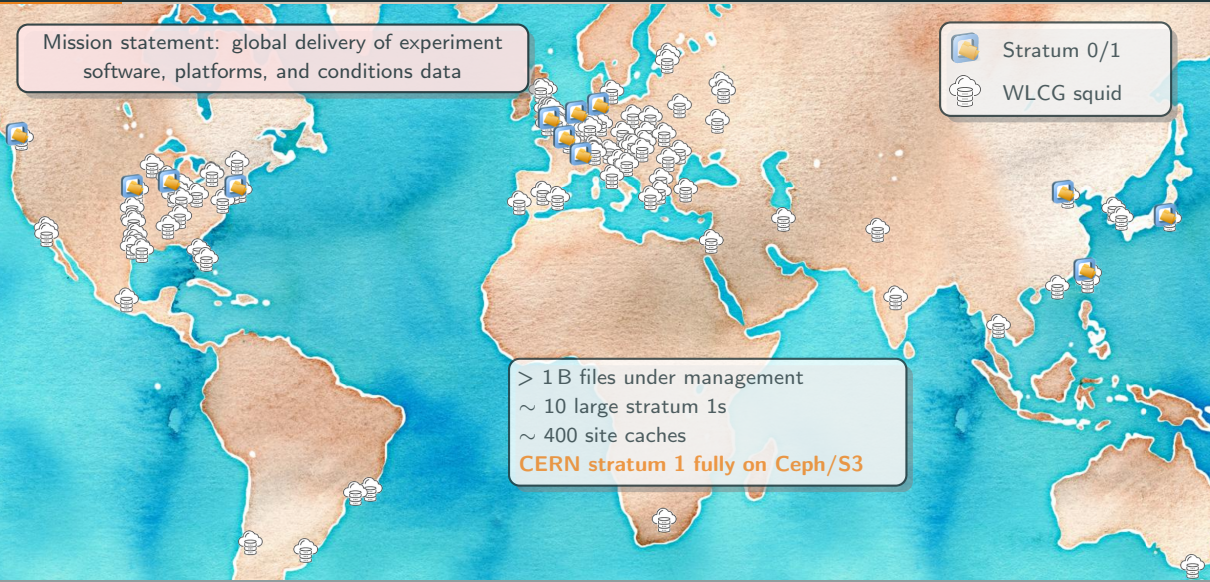
Jakob Blomer (CERN)

EGI Webinar, 15 October 2020

CernVM-FS Worldwide Deployment



CernVM-FS Worldwide Deployment



CernVM-FS Worldwide Deployment

Mission statement: global delivery of experiment software, platforms, and conditions data

Stratum 0/1

WLCG squid

Zoomed into Europe:

Principal Content Types

❶ Production Software

Example: */cvmfs/atlas.cern.ch*

- ✓ Most mature use case
- Containerizing publish workflows

❷ Integration Builds

Example: */cvmfs/lhcbdev.cern.ch*

- ✓ Most challenging application (massive load)
- Continous improvements: parallel publishing, faster GC, shorter propagation

❸ Unpacked Container Images

Example: */cvmfs/unpacked.cern.ch*

- ✓ Enables large scale container deployment
- Integration for most container runtimes

❹ Auxiliary data sets

Example: */cvmfs/alice-ocdb.cern.ch*

- ✓ Benefits from internal versioning
- Depending on volume requires more planning for the CDN components

Current focus of developments

The Grid's Container Hub

/cvmfs/unpacked.cern.ch

- > 500 images
- > 2TB
- > 35M files

/cvmfs/singularity.opensciencegrid.org

- > 500 images
- > 2TB
- > 40M files

A large variety of images readily available to run with singularity on the grid, including **base operating systems**, **experiment software stacks**, **explorative tools (ML etc.)**, **user analyses**, and special-purpose containers such as **folding@home**

```
[jblomer@lxplus.cern.ch]$ singularity exec \
  '/cvmfs/unpacked.cern.ch/registry.hub.docker.com/library/debian:stable' \
  cat /etc/issue
Debian GNU/Linux 10 \n \l
```

Containers and HEP

Why Containers

	Desired Property	Standard Technology	HEP Approach
Isolation	Job X does not impact job Y on the same node	cgroups, namespaces	batch scheduler
Independence	Application brings all its dependencies	chroot, overlayfs	HepOSlibs, CernVM-FS
Distribution	Make software available on the nodes when needed	tar, container registry	CernVM-FS
Orchestration	Schedule distributed jobs on multiple nodes	kubernetes	batch scheduler

We are aiming for the isolation & orchestration capabilies from standard industry tools with the flexibility and efficiency of CernVM-FS based software distribution.

File System Gymnastics

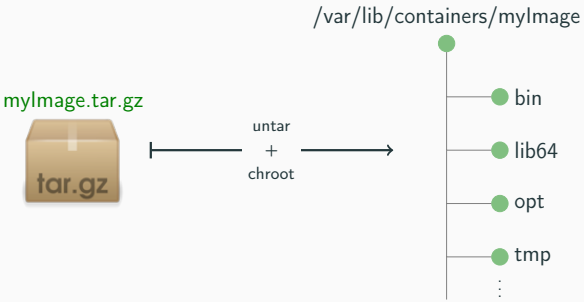


Image **The root file system of the container, either tarball-packed or exploded in a directory**

Layer A part of an image, e.g. only the files of the base operating system

Overlay Stack of layers that create a (writable) root file system

Underlay A writable root file system created from a read-only root file system

File System Gymnastics

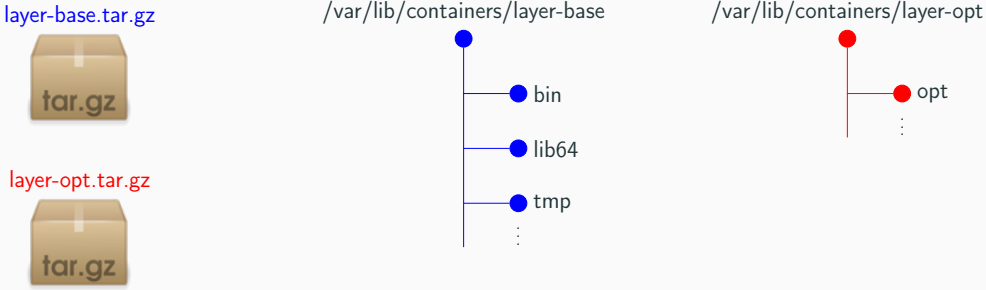


Image The root file system of the container, either tarball-packed or exploded in a directory

Layer **A part of an image, e.g. only the files of the base operating system**

Overlay Stack of layers that create a (writable) root file system

Underlay A writable root file system created from a read-only root file system

File System Gymnastics

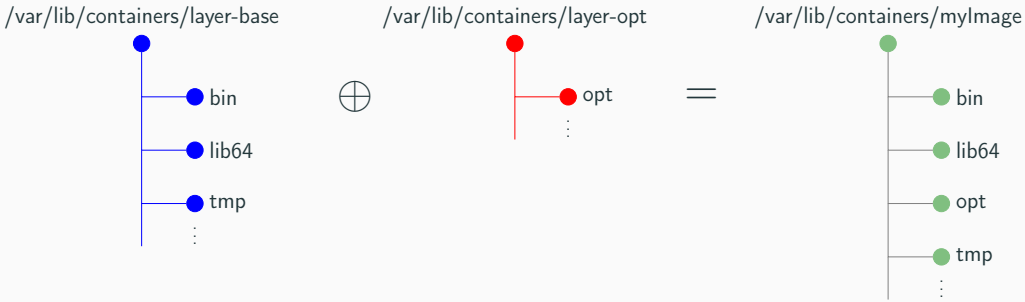


Image The root file system of the container, either tarball-packed or exploded in a directory

Layer A part of an image, e.g. only the files of the base operating system

Overlay **Stack of layers that create a (writable) root file system**

Underlay A writable root file system created from a read-only root file system

File System Gymnastics

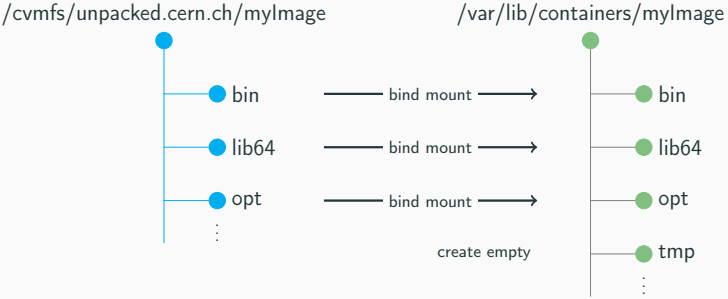


Image The root file system of the container, either tarball-packed or exploded in a directory

Layer A part of an image, e.g. only the files of the base operating system

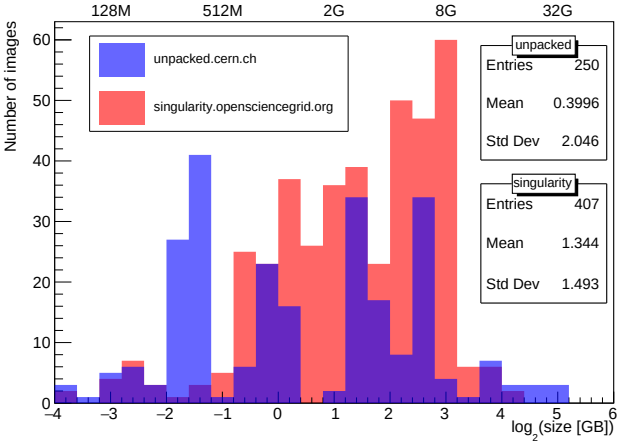
Overlay Stack of layers that create a (writable) root file system

Underlay **A writable root file system created from a read-only root file system**

Container Image Sizes

Distribution of container images sizes in
`/cvmfs/unpacked.cern.ch` and `/cvmfs/singularity.opensciencegrid.org`

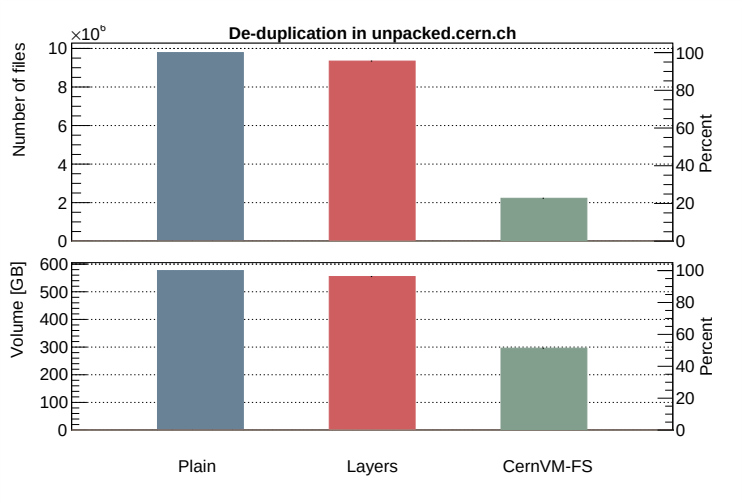
It's easy to overflow the worker node scratch space with multi-gigabyte images



De-duplication

Comparison of de-duplication efficiency between layers and file-based storage (CernVM-FS)

De-duplication works properly only on file-level granularity



Worker Node Software Space

Observations from CERN lxbatch farm:

- Looked at 3 different images (ATLAS and CMS base images)
- Found on > 250 worker nodes
- For each image:
  - 2 % to 9 % of the **image volume** in the worker node cache
  - Site-wide: $\sim$ 15 % of the volume cached
- Not yet including de-duplication effects in the worker node cache

→ **$\times$10 to $\times$50 higher image distribution efficiency from /cvmfs hosted images compared to** `docker pull ...`

Containers and CernVM-FS

Containers and CernVM-FS: Overview

❶ Provide /cvmfs in a container

If /cvmfs is available on the host

- Bind mount from host to container

On opportunistic resources:

- **Unprivileged mounting inside container**
  Uses of user-level fuse mounts (EL >7.8); challenge on sharing the cache among containers
- **As of CernVM-FS 2.8: "service container"**
  CernVM-FS client pre-packaged as a container
- **Pre-mounted by singularity**

❷ Provide container from /cvmfs

Unpacked images on /cvmfs for scalable distribution

Requires:

1. Container image conversion:
   automated with the CernVM-FS DUCC service
2. Container runtime plug-in required for layered images

Reminder:

- "*Flat image*": starts container from unpacked root file system
- "*Layered image*": constructs root file system with Overlay-FS from several directories

Use cases ❶ and ❷ can be combined

Simple Case: CernVM-FS Available on the Host

```
$ docker run -v /cvmfs:/cvmfs:shared busybox ls /cvmfs/sft.cern.ch
README.md lcg
```

```
$ singularity exec -B /cvmfs docker://busybox ls /cvmfs/sft.cern.ch
README.md lcg
```

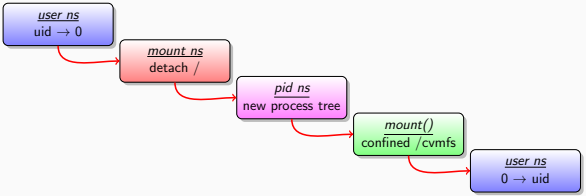
Important: use *shared* bind mount with docker so that that repositories can be mounted on demand from inside the container

Unprivileged Mounting with `cvmfsexec`

```
$ cvmfsexec grid.cern.ch atlas.cern.ch -- ls /cvmfs
atlas.cern.ch cvmfs-config.cern.ch grid.cern.ch
```

Technical foundations

- User namespaces completing container support
- As of Linux kernel version 4.18 (EL8, but also EL 7.8), **fuse mounts are unprivileged in user name spaces**
- Overlay-FS implementation available as a fuse module



CernVM-FS Service Container

- CernVM-FS client in a minimal container
- Mounted /cvmfs inside container can be "leaked" to the outside host
- Alternative to system package based installation, e.g. on container-only operating systems
- Foundation of the kubernetes daemonset deployment ▸ sample deployment

```
$ docker run -d --rm \
    -e CVMFS_CLIENT_PROFILE=single -e CVMFS_REPOSITORIES=sft.cern.ch \
    --cap-add SYS_ADMIN --device /dev/fuse \
    --volume /cvmfs:/cvmfs:shared
    cvmfs/service
$ ls /cvmfs/
cvmfs-config.cern.ch sft.cern.ch
```

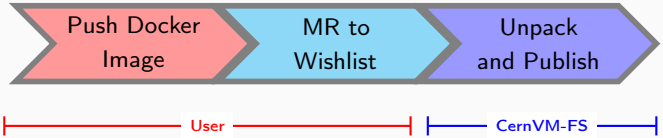
For HPCs: Pre-mounted by Singularity

- With the new Fuse3 libraries, mounting can be handed off to a trusted, external helper.
- Fuse3 libraries have been backported to EL6 and EL7 platforms.
- Gives access to /cvmfs in containers started by singularity (`singularity --fusemount`)
- **Required cvmfs client to be installed and prepared in the container**

```
$ CONFIGREPO=config-osg.opensciencegrid.org
$ mkdir -p $HOME/cvmfs_cache
$ singularity exec -S /var/run/cvmfs -B $HOME/cvmfs_cache:/var/lib/cvmfs \
  --fusemount "container:cvmfs2 $CONFIGREPO /cvmfs/$CONFIGREPO" \
  --fusemount "container:cvmfs2 sft.cern.ch /cvmfs/sft.cern.ch" \
  docker://davedykstra/cvmfs-fuse3 ls /cvmfs/sft.cern.ch
README.md lcg
```

Container Deployment from CernVM-FS

Container Conversion Service



Wishlist https://gitlab.cern.ch/unpacked/sync

```
version: 1
user: cvmfsunpacker
cvmfs_repo: 'unpacked.cern.ch'
output_format: >
  https://gitlab-registry.cern.ch/unpacked/sync/$(image)
input:
   - 'https://registry.hub.docker.com/library/fedora:latest'
   - 'https://registry.hub.docker.com/library/debian:stable'
   - 'https://registry.hub.docker.com/library/centos:*'
```

Multiple wishlists possible, e. g. experiment specific

/cvmfs/unpacked.cern.ch

```
# Singularity
/registry.hub.docker.com/fedora:latest -> \
  /cvmfs/unpacked.cern.ch/.flat/d0/d0932...
# containerd, k8s, podman
/.layers/f0/1af7...
```

Ongoing work on direct registry integration, i. e. `docker push` triggers image conversion (ETA 2021)

Container Runtime Integration

```
[jblomer@lxplus.cern.ch]$ singularity exec \
  '/cvmfs/unpacked.cern.ch/registry.hub.docker.com/library/debian:stable' \
  cat /etc/issue
Debian GNU/Linux 10 \n \l
```

Runtime	Type	CernVM-FS Support
Singularity	flat (+ layers)	**native**
runc	flat (+ layers)	**native**
docker	layers	***"graph driver"* image storage plugin**
containerd / k8s	layers	**pre-release** ▸ remote snapshotter
podman	layers (+ flat)	**pre-release** ▸ pull request

Currently improving documentation, examples, integration tests for different deployment options
→ `https://cvmfs.readthedocs.io/en/latest/cpt-containers.html`

Example for Large-Scale Deployment: Folding@Home

Runs on the grid off containers served from /cvmfs

Ephemeral Publish Container

Ephemeral Publish Container

- A new command, `cvmfs_server enter`, creates a sub-shell with a writable /cvmfs
- Uses internally user namespaces and fuse-overlayfs
- Works unprivileged on any modern Linux that can mount the client
- Meant to be used on build nodes
- **Ongoing work, to be released in CernVM-FS 2.8** pull request

```
$ cvmfs_server enter hsf.cvmfs.io
...Opens a shell with write access to /cvmfs/hsf.cvmfs.io
$ cvmfs_server diff --tarball | gzip > changes.tar.gz
...Back to read-only mode
```

Summary

Announcements

EGI Clinic on HT-Condor and CernVM-FS

3 November 2020

https://indico.egi.eu/event/5000/sessions/4508/

CernVM Workshop 2021

1–3 February 2021

https://indico.cern.ch/e/cvm21

Summary & Outlook

A container strategy for the grid:

1. Exploit isolation and orchestration capabilities from standard tools (singularity, k8s, . . .)
2. Exploit distribution efficiency from CernVM-FS

There is a continuous effort from the CernVM-FS team to integrate well with container runtimes

Possible future landscape of containerized software stacks

Production workflows (e. g. simulation)	Explorative workflows (e. g. testing ML tools)
Full app universe provided by /cvmfs	Full container deployed from /cvmfs
Minimal Compatibility Container glibc + ε	

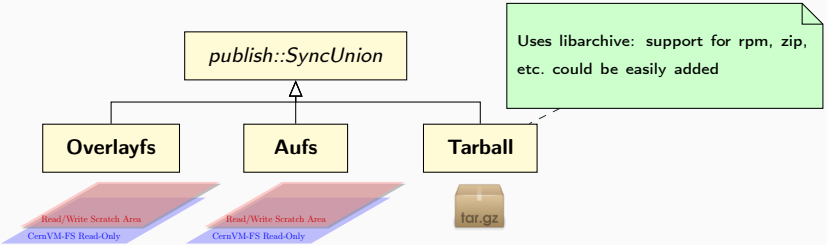
Backup Slides

Enabling Feature for Container Publishing: Tarball Ingestion

Direct path for the common pattern of publishing tarball contents

```
$ cvmfs_server transaction
$ tar -xf ubuntu.tar.gz
$ cvmfs_server publish
```

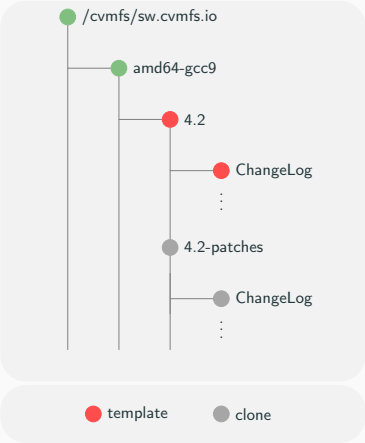
```
$ cat ubuntu.tar.gz | \
    cvmfs_server ingest -t -
```



Performance Example

Ubuntu 18.04 container – 4 GB in 250 k files: **56 s untar + 1 min publish** vs. **74s ingest**

Template Transactions



Pull request

```
cvmfs_server transaction \
  -T /amd64-gcc9/4.2=/amd64-gcc9/4.2-patches \
  sw.cvmfs.io
```

- As part of opening the transaction, "4.2" is cloned to "4.2-patches"
- Meta-data only copy, thus extremly fast
- Only changes on top need to be published