CVMFS Image Ingestion
&
Container Runtime plugin
...

pre-GDB meeting 05/05/2020

Simone Mosciatti

Unpacked.cern.ch - structure

- Visible, directories of symlink to access flat root-filesystem of container images

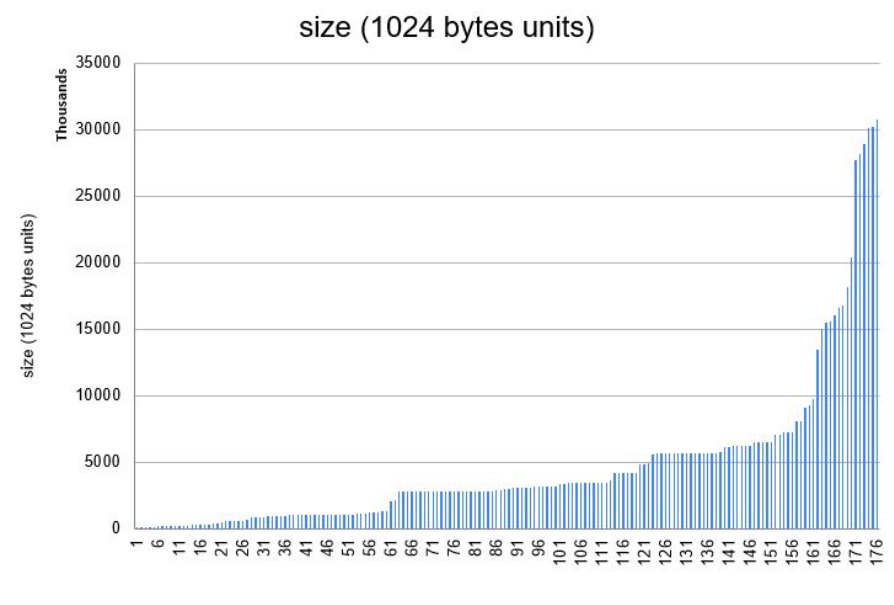
- Hidden directories to store the actual flat filesystems
- Hidden directories to store layers of the images
- Hidden directories to store metadata

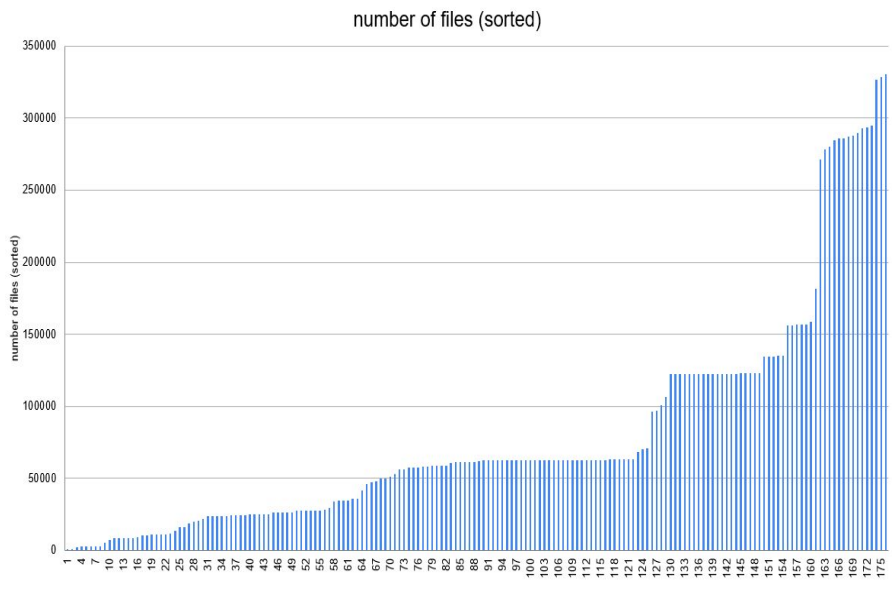
```
/cvmfs/unpacked.cern.ch/registry.hub.docker.com/atlas/athena:22.0.9 -> ../../.flat/8a/8a85...3fdd
```

unpacked.cern.ch stats

- Images: ~175 active (250 repeated)
- Total Size: ~1665G (818G of unique flat images)
- Storage Size: ~155G
- Deduplication: 155/1665 = 9%
- Total files: 22M
- Total objects: 2M

Images and size distributions





Status - platforms ready today

- singularity
- Docker thin-images

Status - singularity

- Simple and well tested in production

```
$ /cvmfs/unpacked.cern.ch/util/bin/singularity exec \
> /cvmfs/unpacked.cern.ch/registry.hub.docker.com/atlas/athena:21.0.23
```

- It is running the COVID-19 fold@home CERN contribution
- Of course runs mainly HEP tasks

Status - Docker thin-images

- Requires the graphdriver plugin installed
- Need to use thin-images produced by DUCC instead of the regular Docker images
- Documentation and details

Future platforms

1. containerd
    - docker
    - k8s
2. podman

Future platform - containerd

- Under the hood engine for docker and k8s
- Most of the work done by NTT Japan (ktock)
- Fruitful discussion between CERN and containerd developers
- Based on the remote snapshotter idea

Future platform - containerd roadmap

- Prototype worked quite well (~6 months ago)
  - Very simple implementation
- All piece are about ready, but it needs some polishing
- Need one last push to make it work for us
  - Code in containerd master
  - Need releases
  - Need k8s to pick it up

Future platform - podman

- Docker replacement from RedHat (IBM)
  - `alias docker=podman`
  - Default on CentOS8
- Run without sudo
  - Maybe suitable for GRID use (some issue with file permission)

Future platform - podman roadmap

- GSoC approved project for running docker images fetching the layers from CVMFS
- Interesting discussion/issue to run a podman container directly from an unpacked filesystem (like singularity) from a read-only directory.

Ingestion of images in CVMFS

- Working solution
  - DUCC
- Future plan
  - Docker registry shim

Ingestion of images in CVMFS - DUCC

- Based on concept of wish-list
  - Users / experiments specify what images they want
  - The get pulled, unpacked and ingested into CVMFS
  - Unpack both layers and full filesystem
- Support of `*` in image tag
- Rather stable
- Generally rather idle, margin to ingest more images
- Spiky workload, lead to high latency for ingestion
  - Unfortunately no simple solution here

We are considering adding interactive feedback to let the user follow the ingestion process.

Ingestion of images in CVMFS - Docker registry shim

- Expose same interface than a docker registry
- Accept push from docker clients
- Wait until the image is completely ingested in CVMFS
- Return to the client
- Need to address client timeout (e.g. retry, publish during throttled upload)

- Promising prototype
- Issue on how to create the unpacked whole filesystem
- Concurrency issue with DUCC
- Nothing impossible to solve, but requires thoughts and time

Closing remarks

- Several parallel development lines to improve container runtimes integration and to speed up and simplify the ingestion model
- We prioritize the development according to the needs and feedback
- Let us know what are the most urgent needs
- Feedback could be louder