# Containers WG status

GDB 12 September 2018

Gavin McCance

# Recent meetings

- Last GDB update was January this year
- There have been 4 meetings since then
- We had foreseen more frequent meetings, but developments didn't really justify it
- Focus still very much on common Singularity deployment and usage
    - Reminder that CMS have required sites to have Singularity since Q1 2018

# Singularity what

- Provides simple command line to run your job inside a containerised environment
- Isolation for multiple payloads of the same pilot
- Separation of job's OS from system one. We typically mount the container OS from an unpacked directory in CVMFS

# Packaging update

- Previous EPEL maintainer was maintaining a lot of unreferenced patches and was slow to respond, so we were maintaining a version in the WLCG repo and asking sites to use that

- Brian and Dave took over as maintainers

- Unreferenced patches were removed since 2.5.2

- Some patches useful to WLCG (referencing upstream PRs) are being maintained in the EPEL version. Later...

- Sites should use the EPEL version, currently 2.6.0-1.1
  (it's been removed from the WLCG repo)

# Bind mounts

- Bind-mounting is necessary to share selected files (jobdir and local storage) with jobs running inside of Singularity containers but require that the mountpoints pre-exist in the image
    - Fine for CMS, who bind-mount their jobdir onto /srv, which is in the image
    - Not fine for ATLAS who, at some sites, need to bind-mount their storage caches onto an arbitrary site-dependent directory
- In principle, an overlay filesystem solves this, but:
    - Only available on CC7
    - Fails when Singularity images are distributed unpacked in filesystems that use extended attributes, including CVMFS
    - Requires Singularity run privileged (setuid), even on CC7, to mount the overlay filesystem. Supporting sites to be able to run unprivileged is a major goal of the working group

# "Underlay" feature

- Proposed by Vincent and implemented by Dave
- Uses the following steps:
    - In a writeable scratch space, make caller-requested paths plus every other unhidden path from the OS image
    - Bind mount the writeable space read-only to a new place that will become the root of the container
    - Bind-mount everything on the read-only root
- Assuming availability of user-space mount namespaces, should allow both Atlas and CMS use unprivileged Singularity on CC7
- Happily, Redhat just announced full support for unprivileged user namespaces in 7.6

# "Underlay" status

- Upstream pull-request there
- Dave is currently maintaining these PRs in EPEL version, since 2.6.0-1.1
- Singularity upstream are unlikely to take this (2.6 is the last). Instead are concentrating their energy on their 3.0 Golang rewrite, which does contain this feature
- ATLAS has been testing and iterating the underlay feature with the EPEL version
    - CMS and LIGO also expect to be able the benefit from underlay in the future

# Current goals

- New underlay feature looks helpful to advance the option of having unprivileged usage of Singularity
    - Experiments are testing
        - We should now be able to start converging on a common baseline for Singularity configuration
- WG still focussing on the 90% baseline solution for "standard sites", recognising that there are other sites, such as HPC ones, that will need more work
- Current version is from EPEL, 2.6.0-1.1

# Future

- New Singularity v3 rewrite from Sylabs upstream should also have the features we need, including "underlay"
    - To be tested and security-reviewed once we have it (1st alpha available now)
        - The security review is being arranged via https://trustedci.org/application/ with Sylabs taking the lead
- Questions on list regarding more mainstream "OCI" container technology (libcontainer, runc, ...). They all require unprivileged user namespaces.
    - Assuming we want a common interface across SL6/CC7 Singularity is still the thing. When the need for SL6 fades away, it's potentially worth a revisit

# Other things to discuss: images

- (Un)packed image distribution, in CVMFS
  - Can we converge on a common base image from which the experiments can build?
  - Can we agree on a common CVMFS repo? Is it even desirable?
  - Can we agree on a common workflow / tooling?