# Singularity in CMS

“Over a million containers served”

# Introduction

- The topic of containers is *broad*!
  - I’m filtering out a lot of relevant details, particularly **why** we are using Singularity and e.g., not Docker.
  - We’re big Docker users locally: **Nebraska runs Singularity inside Docker**. 100% of pilots go inside Docker here!
- I’m also taking the **CMS-centric view**, even though work was done by many organizations.

# What problems are we solving?

- Simple **isolation**: Protect pilot from payloads and payloads from each other. Specifically:
  - *File isolation*: pilot determines what files the payloads can read and write.
  - *Process isolation*: payload can only interact with (see, signal, trace) its own processes.
  - There are other kinds of isolation (e.g., resource management, kernel isolation, network isolation) that are useful *but not required*.
- **glexec replacement**: Retire our particularly problematic current solution to isolation.
- **Homogeneous / portable OS environments**: Make user OS environment as minimal and identical as possible

# What is Singularity?

- Singularity is a container solution tailored for the HPC use case.
  - It allows for a portable of OS runtime environments.
  - It can provide isolation needed by CMS.
- Simple isolation: Singularity does not do resource management (i.e., limiting memory use), leaving that to the batch system.
- Operations: No daemons, no UID switching; **no edits to config file needed**. "Install RPM and done."
- Goal: User has no additional privileges by being inside container. E.g., disables all `setuid` binaries inside the container.

http://singularity.lbl.gov

# Singularity and CMS

- **Singularity meets the CMS needs** for isolation! WLCG Isolation and Traceability Task Force adopted it as the replacement technology for glexec.
- It also solves a sticky problem for CMS: provides a portable OS environment.
  - CMS *cannot* run its RHEL6 binaries inside a RHEL7 environment. Hence, **CMS *cannot* transition to RHEL7** using our traditional techniques.
  - Using container technologies means the *payload* can run in an arbitrary OS environment, different from the pilot. In fact, the **pilot could start 8 payloads inside 8 different Linux distributions** if it wanted.
- **Sites may provide a RHEL7 environment to the pilot *only if* they also provide `singularity`**. Otherwise, CMS may be unable to utilize the site.
  - Goal was have this ready by April 1. Still smoothing out bugs.
  - Have 3 sites that have default pilot using Singularity instead of glexec.

# Singularity Integration

- To use Singularity, we need a few things.
- Available at sites:
    - Given it is popular at many HPC centers, **several OSG sites already had it** installed.
    - Available from EPEL, but EPEL version is too old for our use.
    - In ~ November, got permission from OSG Security to ship it in OSG Upcoming repository. Done as of January 2017.
    - **Long-term goal remains to utilize version from EPEL**. Currently in WLCG repo!
- Integrate with pilot infrastructure:
    - HTCondor can invoke Singularity directly or Singularity can be integrated into the wrapper script.
    - Since there is no separate daemon or UID switching, no code needs to be changed besides job startup! For OSG & CMS, this was about 400 lines of bash.

# Portable OS environment

- How do we deliver an OS environment to CMS pilots?
  - Singularity has its own image creation utilities *or* **can convert Docker images**.
    - Given the immense ecosystem of Docker images and tooling, we have chosen the latter approach.
  - Traditionally, Singularity images are a single file. These get large: simple LIGO image might be about 4GB. Singularity can also just read from a directory.
  - What tool would CMS use to distribute a directory of software across the global infrastructure? CVMFS
  - CVMFS also provides per-file caching and file-level de-duplication. To launch python only requires downloading 3MB of data from a 3GB image. CVMFS also **provides efficient cache management**.

# Inside the CMS container

- Inside the container, we have:
  - User payload processes, running (real UID) as the pilot user.
  - A full copy of the base RHEL6 (or 7!) OS, served from CVMFS.  By default, everything is *read-only*.
  - Generate basic `passwd`, `group`, and `resolv.conf` so user environment is relatively sane.
  - User working directory is bind-mounted to `/srv`. `$HOME` is set to `/srv`.
    - CVMFS and any POSIX storage elements are also a bind-mount inside the container.
    - User environment is updated to correct any changed file paths.
  - Pilot can select other files to copy or bind-mount inside container.

# SAM Tests

- Currently, glexec is a required SAM test:
  - As a security measure, Singularity disables all setuid activities in the image: no setuid => no glexec.
  - We have a separate SAM test for singularity functionality.
  - Working to correctly “OR” results together of both tests.
- This will be ongoing; many of our tests will need tweaks to successfully land in RHEL7.

# Get your site ready!

- We can invoke singularity as soon as the RPM is installed - no config file tweaks needed!
    - Sadly, this does not imply we can successfully run CMS jobs.
    - Must find all your site's POSIX filesystem dependencies; what needs to appear in the container?
- CRAB3 won't match slots that use Singularity but are RHEL7: **bug fix, not deployed**.
- Things to check:
    - Are SITECONF files kept in CVMFS?
    - POSIX storage?
        - Does CMS bind-mount your POSIX storage inside the images?
        - No user switching: Can the pilot read from the CMS directories?
        - No user switching: No stageout from image to the site storage.
    - May need to land SAM tests on RHEL6.

# Traceability

- glexec provides traceability *and* isolation; singularity provides *only* isolation. **How do we provide traceability?**

- Solution 1: VO maintains sufficient records. Site contacts the VO when they need to trace specific activity.

  - Site puts significant trust in the VO and its record keeping; most sites are OK with this.

# Traceability - Site Level

- Alternately, for each payload launch, the VO can assert to the site what user it will run.
    - Effectively we do this with glexec when the VO sets GLEXEC_CLIENT_CERT to point at the user's cert.
- Current approach (HTCondor-CE sites): pilot can callout to HTCondor-CE and advertise the currently running payloads.
    - This already exists. Extending HTCondor-CE to record this data into an audit log.
    - Expect delivery of feature in May or June 2017; requires patches to HTCondor and HTCondor-CE.

# Conclusions

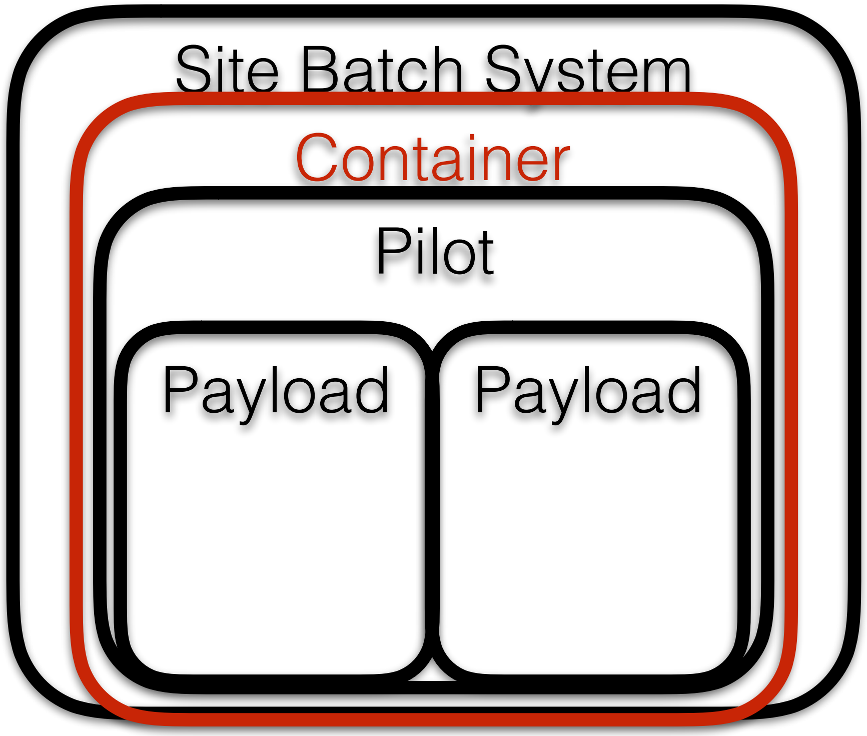
- Singularity meets an important CMS need; a few brave sites are in production.
- Sites may be able to decommission `glexec` as soon as they deploy singularity. Nebraska will hopefully do this in April!
  - *Greatly* simplifies the AAI setup: removing all pool accounts from the site.
- Delivers a compelling feature set beyond LHC. Both isolation and portable user environments.
- Looking for interested sites to participate!
  - It's an exciting-but-young effort: there will be some speed bumps, but will benefit from your help!

# Backup Slides

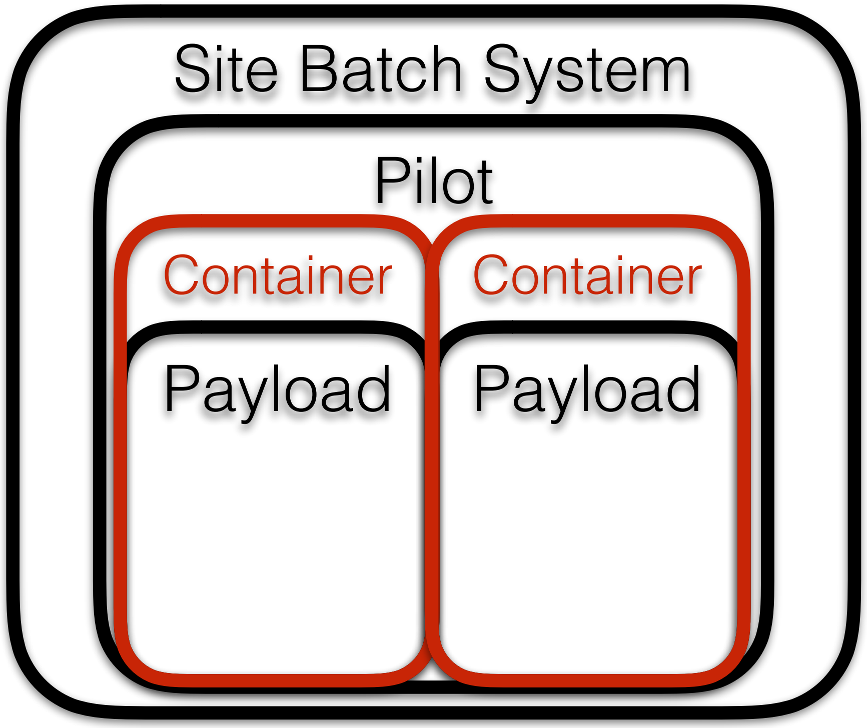
# **Who** is in a container?

- Three options when using containers:
  - A: Batch system starts pilot inside a container.
  - B: Pilot starts each payload inside its own container.
  - C: Combine A and B.
- Option A does not meet our isolation goals. **Option B does**.
- It is important to allow sites to do their container work: ***must* keep option C viable**!

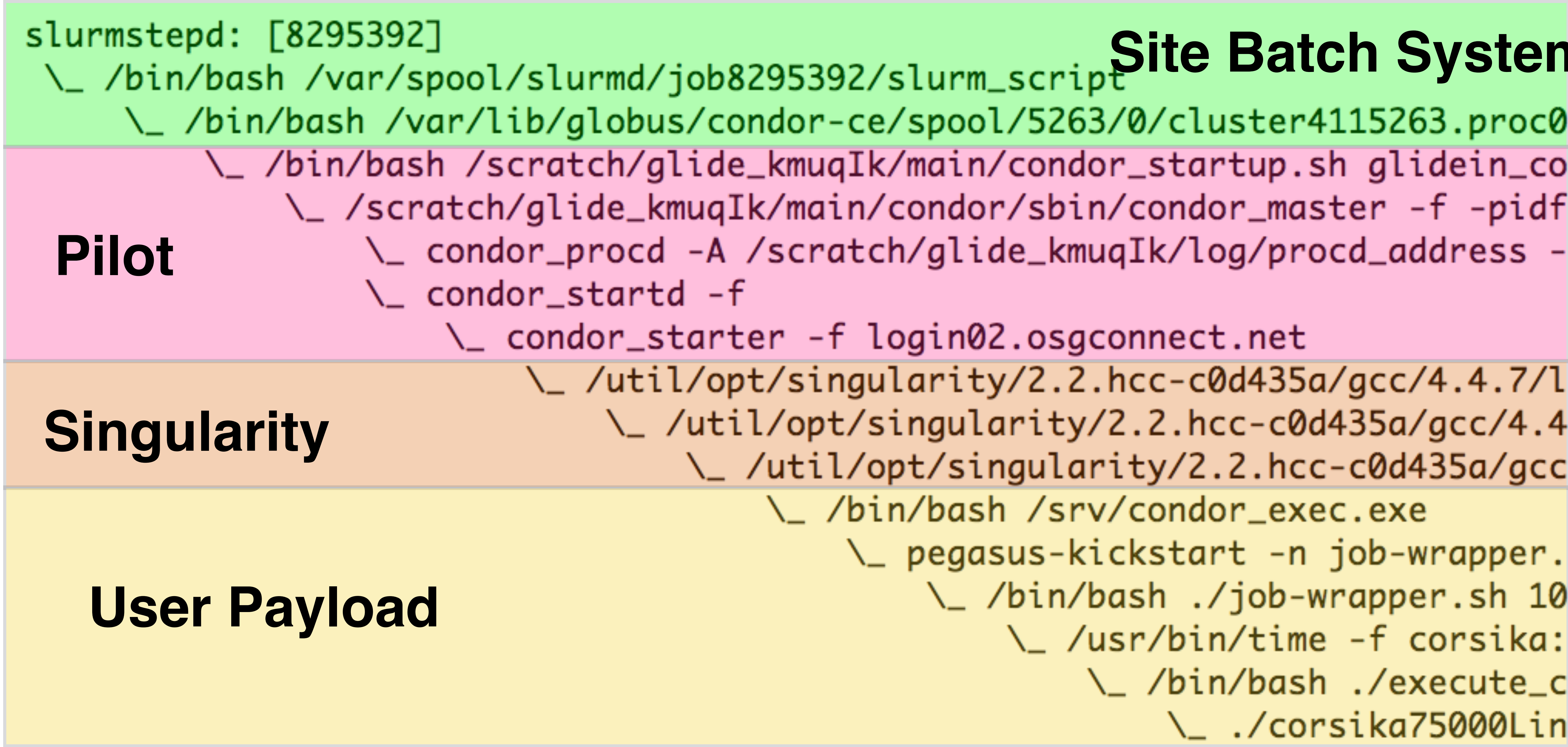
Option A:



Option B:



# View From the Worker Node



```
slurmstepd: [8295392]
 \_ /bin/bash /var/spool/slurmd/job8295392/slurm_script
     \_ /bin/bash /var/lib/globus/condor-ce/spool/5263/0/cluster4115263.proc0
         \_ /bin/bash /scratch/glide_kmuqIk/main/condor_startup.sh glidein_co
             \_ /scratch/glide_kmuqIk/main/condor/sbin/condor_master -f -pidf
                 \_ condor_procd -A /scratch/glide_kmuqIk/log/procd_address -
                 \_ condor_startd -f
                     \_ condor_starter -f login02.osgconnect.net
                         \_ /util/opt/singularity/2.2.hcc-c0d435a/gcc/4.4.7/l
                             \_ /util/opt/singularity/2.2.hcc-c0d435a/gcc/4.4
                                 \_ /util/opt/singularity/2.2.hcc-c0d435a/gcc
                                     \_ /bin/bash /srv/condor_exec.exe
                                         \_ pegasus-kickstart -n job-wrapper.
                                             \_ /bin/bash ./job-wrapper.sh 10
                                                 \_ /usr/bin/time -f corsika:
                                                     \_ /bin/bash ./execute_c
                                                         \_ ./corsika75000Lin
```

**Site Batch Syste**

**Pilot**

**Singularity**

**User Payload**