Fermilab | U.S. DEPARTMENT OF ENERGY | Office of Science

FIFE and Containers

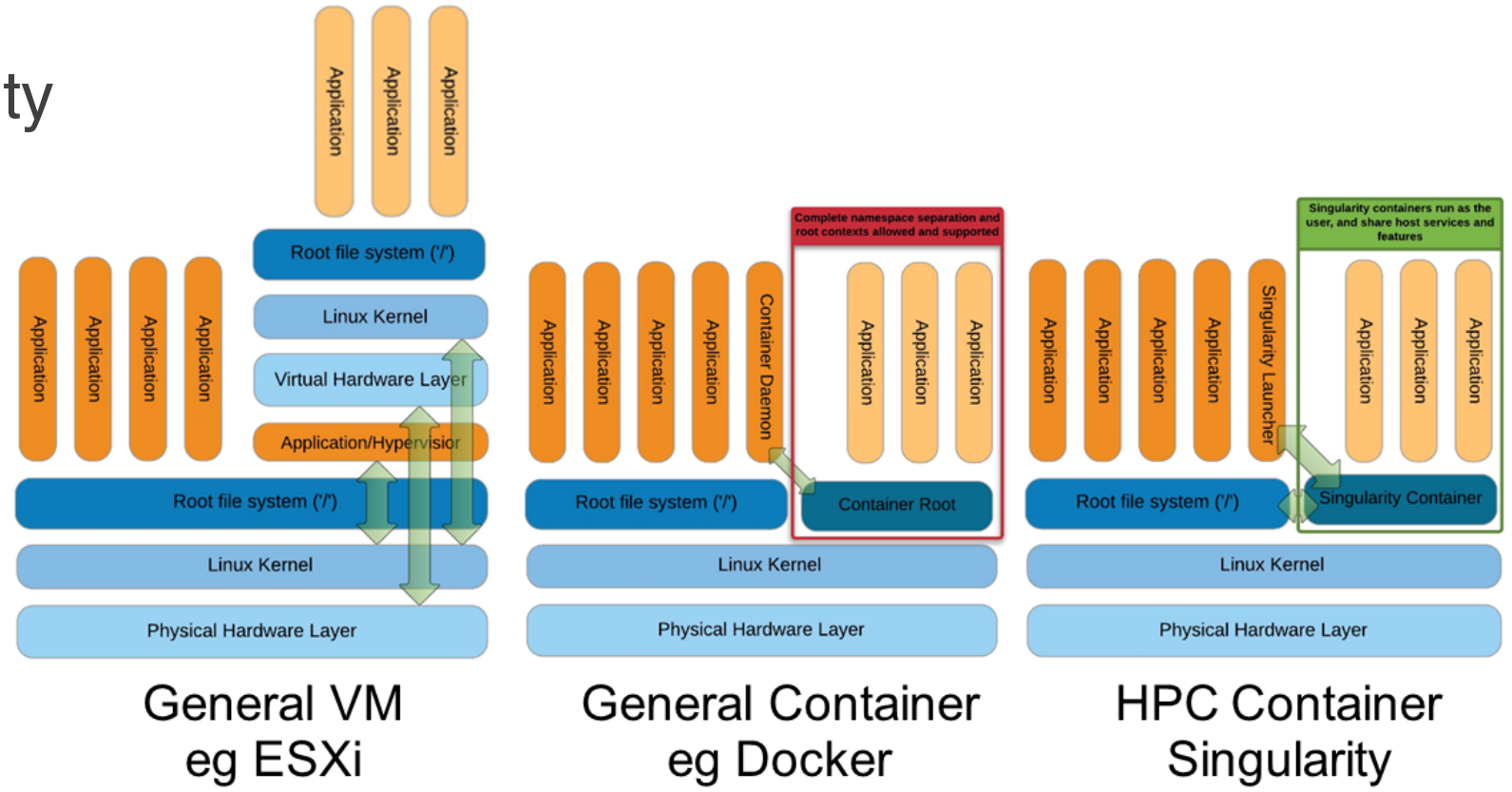
Marco Mambelli

Fermilab

March 28, 2019

Containers

- Virtual Machines (Clouds, AWS, GCE, Azure)
- Docker (microservices)
- Shifter
- Singularity
- gLExec



[Greg Kurtzer keynote at HPC Advisory Council 2017 @ Stanford]

Motivations for Containers

- **Consistent environment (default images)** - If a user does not specify a specific image, a default one is used by the job. The image contains a decent base line of software, and because the same image is used across all the sites, the user sees a more consistent environment than if the job landed in the environments provided by the individual sites.

- **Custom software environment (user defined images)** - Users can create and use their custom images, which is useful when having very specific software requirements or software stacks which can be tricky to bring with a job. For example: Python or R modules with dependencies, TensorFlow

- **Enables special environment such as GPUs** - Special software environments to go hand in hand with the special hardware.

- **Process isolation** - Isolates the job environment, so that a job can not peek at other jobs.

- **File isolation** - Isolates the job file system, so that a job can not peek at other jobs' data.

Motivations for Singularity

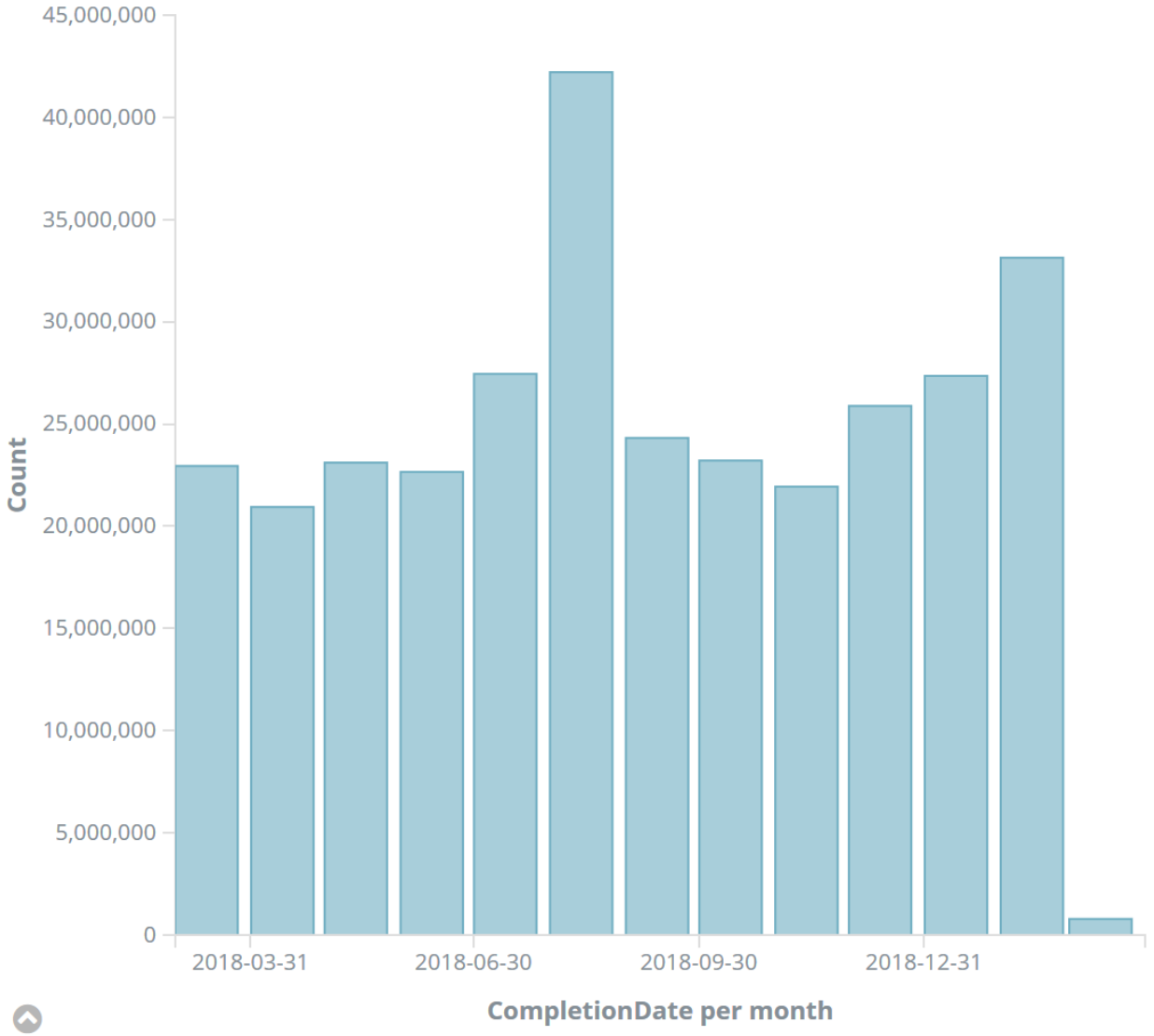
- **Invoked by unprivileged user** - HTC uses unprivileged “pilot jobs” to bootstrap, and those each read from a per-Virtual Organization (VO) queue of jobs from different users.

- **OSG and the science community are using it** – You can benefit of the existing infrastructure.

Container Lifecycle (Hint: ephemeral)

- Every job is encapsulated in a separate container instance
- Container instance dies when the job finishes
- Lots of container image reuse, as workloads generally use one or a small number of images for a large number of jobs
- Application software is mostly outside of the container

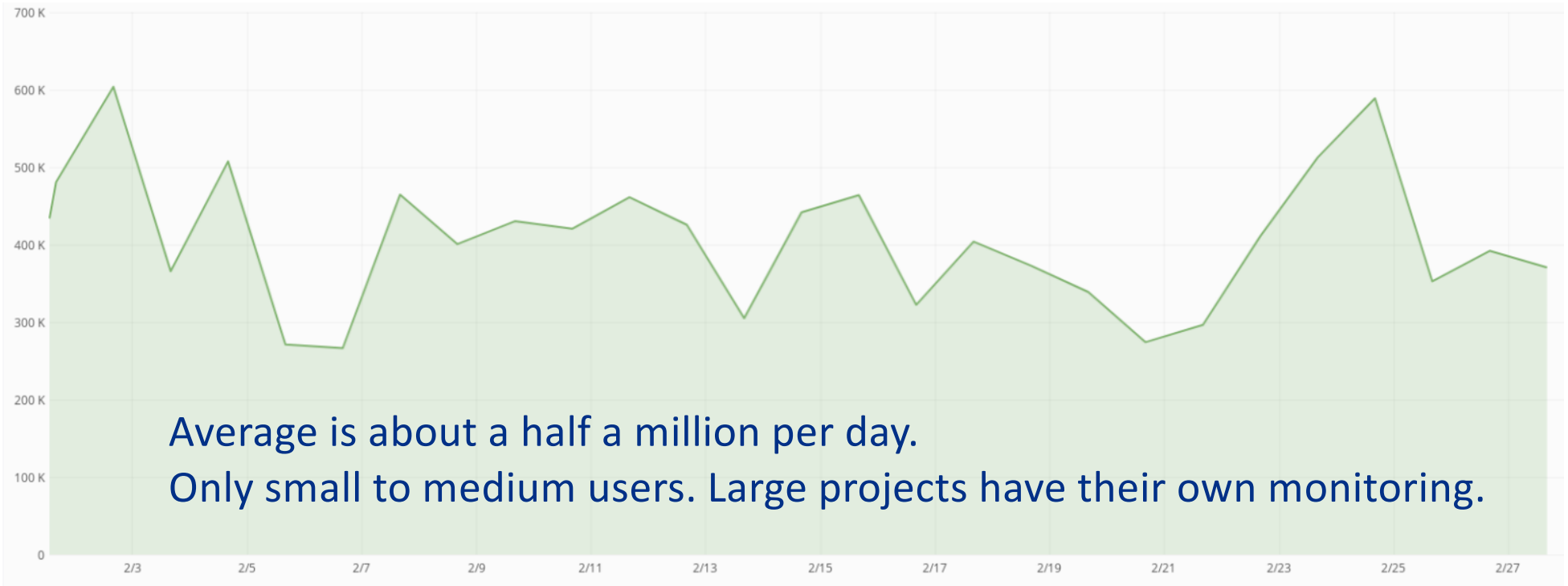
CMS Singularity instances per month



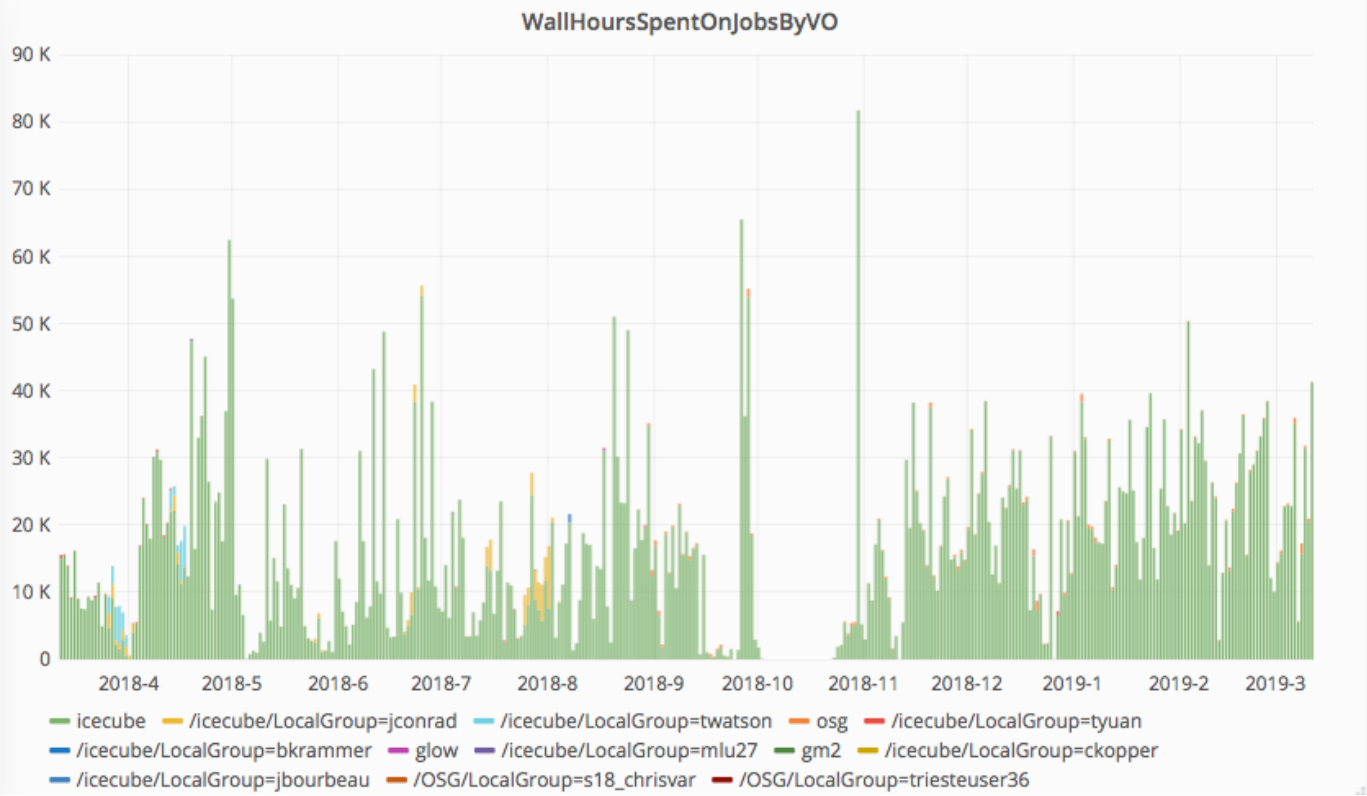
CMS requires all its sites to install Singularity. They do about a million container starts per day worldwide.

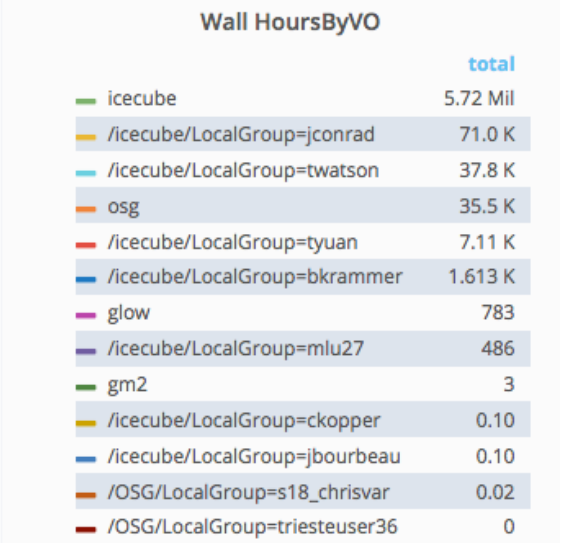
Fermilab

OSG Singularity instances per day



GPU Usage in OSG powered by Singularity





Usage over the last year

Average about 1K cores

Fermilab

CVMFS - CERN Virtual Machine File System

*"The CernVM File System provides a scalable, reliable and low-maintenance software distribution service. It was developed to assist High Energy Physics (HEP) collaborations to deploy software on the worldwide-distributed computing infrastructure used to run data processing applications. CernVM-FS is implemented as a **POSIX read-only file system** in user space (a FUSE module). Files and directories are hosted on standard web servers and mounted in the universal namespace **/cvmfs**."*

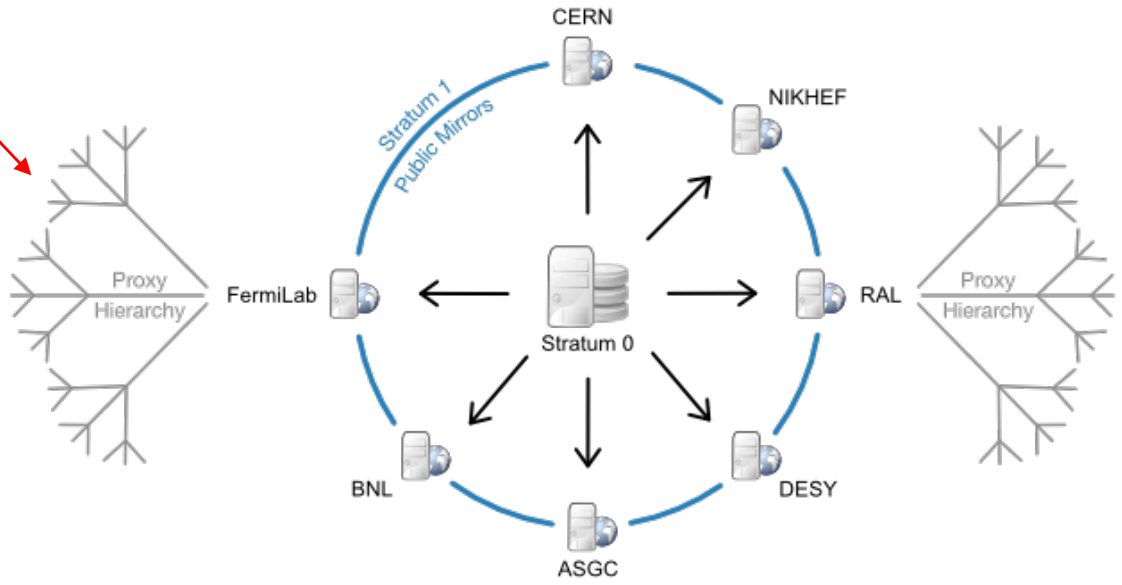
Your job is here!

Pre-existed use of containers

Used for software and some data

Heavily cached, read-only

Available across OSG, EGI, some XSEDE resources



Fermilab

CVMFS features

- **Files appear immediately present, but are only downloaded on demand**
- **Metadata operations are done on the worker node, on "catalogs" downloaded in chunks of about 100K files or less**
- **Files stored and transferred named by secure hash of contents of files, deduplicated and compressed**
- **Cryptographically verified with a digital signature on one small file**
- **Larger files broken up into ~8 MB chunks (by default) to smooth out load on servers**
- **Served from small number of worldwide "stratum 1" servers, cached in http proxy caching servers (squid) at each site, and cached on each worker node** - caches greatly reduce latency and bandwidth
- **Optional "external data server" mode** - for data (metadata uses standard path) of partially reused data files, using separate caching servers at geographically distributed high-capacity network sites

Example CVMFS Repositories

```
/cvmfs/
      ams.cern.ch
      atlas.cern.ch
      cms.cern.ch
      connect.opensciencegrid.org
      icecube.opensciencegrid.org
      fermilab.opensciencegrid.org
      ligo.opensciencegrid.org
      ligo-containers.opensciencegrid.org        <- large project with their own containers
      nova.opensciencegrid.org
      oasis.opensciencegrid.org
      singularity.opensciencegrid.org       <- general containers (next few slides)
      stash.osgstorage.org             <- ~1PB of user published data
```

cvmfs-singularity-sync

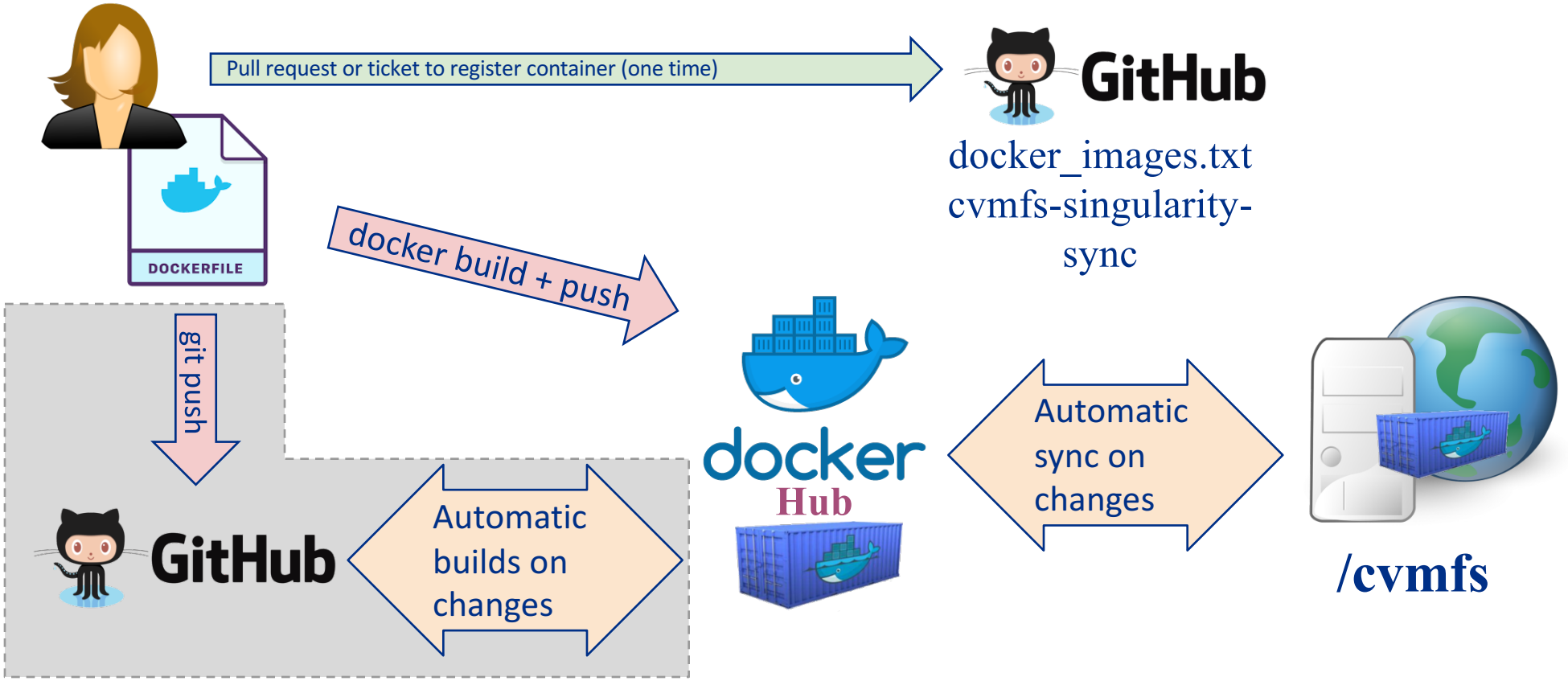
- Containers are defined using Docker
  (see Tony's presentation for the Fermilab infrastructure)
  - Public Docker Hub
- … and executed with Singularity
  - No direct access to the Singularity command line - that is controlled by the infrastructure
- https://github.com/opensciencegrid/cvmfs-singularity-sync

Available Containers

	Image Location	Definti on	Description
EL 6	/cvmfs/singularity.opensciencegrid.org/opensciencegrid/osgvo-el6:latest	GitHub	A basic Enterprise Linux (CentOS) 6 based image. This is currently our default image
EL 7	/cvmfs/singularity.opensciencegrid.org/opensciencegrid/osgvo-el7:latest	GitHub	A basic Enterprise Linux (CentOS) 7 based image.
Ubuntu Xenial	/cvmfs/singularity.opensciencegrid.org/opensciencegrid/osgvo-ubuntu-xenial:latest	GitHub	A good image if you prefer Ubuntu over EL flavors
Ubuntu 18.04 (Bionic)	/cvmfs/singularity.opensciencegrid.org/opensciencegrid/osgvo-ubuntu-18.04:latest	GitHub	A good image if you prefer Ubuntu over EL flavors
TensorFlow	/cvmfs/singularity.opensciencegrid.org/opensciencegrid/tensorflow:latest	GitHub	Base on the TensorFlow base image, with a few OSG package added
TensorFlow GPU	/cvmfs/singularity.opensciencegrid.org/opensciencegrid/tensorflow-gpu:latest	GitHub	Used for running TensorFlow jobs on OSG GPU resources

~150 images, consisting of pre-defined ones by OSG staff, base images from Docker (different OSes, Python, r-base, …) and custom images by our users

User-defined Container Workflow



Extracted Images

OSG stores container images on CVMFS in extracted form (Singularity calls this "sandbox" mode). That is, we take the Docker image layers or the Singularity img/simg/sif files and export them onto CVMFS. For example, ls on one of the containers looks similar to ls / on any Linux machine:

```
$ ls /cvmfs/singularity.opensciencegrid.org/opensciencegrid/osgvo-el7:latest/
cvmfs  host-libs  proc  sys  anaconda-post.log     lib64
dev    media      root  tmp  bin                  sbin
etc    mnt        run   usr  image-build-info.txt  singularity
home   opt        srv   var  lib
```

Result: Most container instances only use **a small part** of the container image **(50-150 MB)** and that part is **cached** in CVMFS! We don't care about docker layers because cvmfs deduplicates everything.

Mountable image files vs CVMFS images

- When a high speed local mounted filesystem is available, singularity mounts the image file on the client as a loopback filesystem – this moves the metadata operations to the client and only reads the pieces actually used – CVMFS does too
- The difference is that files published in CVMFS are instantly available worldwide
- But, most HPC admins are suspicious of FUSE, so we build image files to use HPC allocations
  - CMS and ATLAS application software is much larger than the OS code that they usually have in a container image; normally we bind mount /cvmfs into the container for application code
  - They find it very difficult to trim the size down, so the typical CMS and ATLAS HPC image size is around 200 GB and takes about 8 to 12 hours to build reading from CVMFS, then needs to be uploaded to each HPC filesystem
  - Typically these containers include the “pilot”, so we get no isolation between users and so have to run only “production” jobs all by the same user id, no user analysis jobs

Unprivileged, non-setuid Singularity

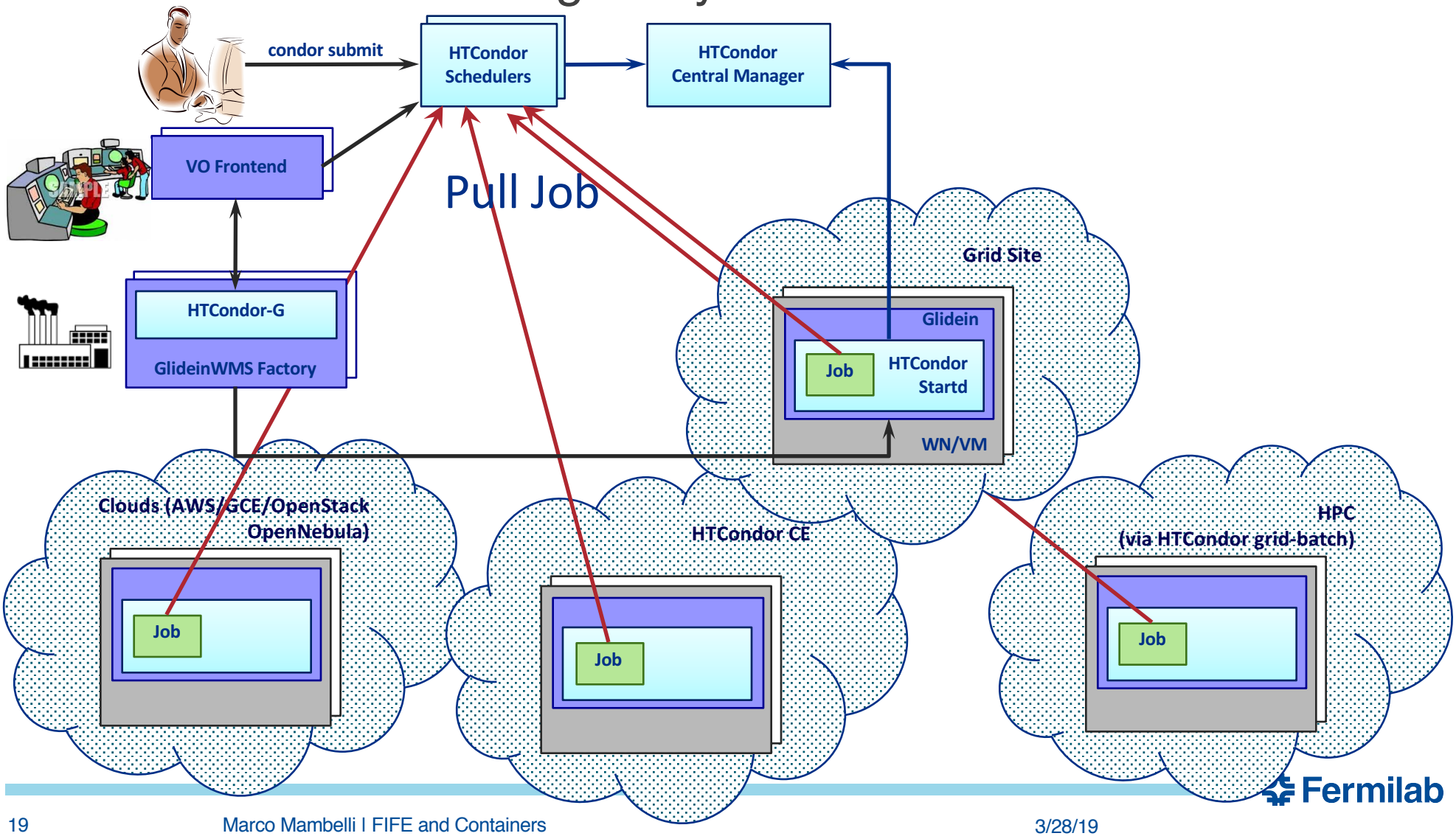
- Because HTC does not need to use loopback mounts, and we can avoid overlayfs by using the underlay feature for adding bind mounts, so we can use the unprivileged namespace feature now standard in RHEL 7.6
  - This is considered to be key for reducing vulnerability to risks of setuid-root. Unprivileged namespaces also get CVEs, but there are many more people examining the Linux kernel than Singularity.
  - Underlay works by first bind-mounting everything requested onto a scratch area, then bind-mounting everything else from the image onto the same scratch area, and using a read-only bind-mount of the scratch area as '/' for the container.
- The ability to mount FUSE filesystems in unprivileged namespaces is now in the latest Linux kernels, which when it becomes available should provide even more highly useful functionality to unprivileged Singularity
- With unprivileged Singularity you can run condor_ssh_to_job to troubleshoot your jobs

Singularity

- Singularity runs a container as a system user
- On RHEL 7.4 and above, enabling unprivileged user namespaces, can run as unprivileged user (there is a version distributed in OASIS)
- Allows to select the namespace isolation (mnt, pid, ipc, net, user)
- Allows to run a different OS
- Is accepted on HPC resources and on other clusters (Docker is not)

GlideinWMS manages the pilots (Glideins) to run jobs

- Used in OSG, CMS and FIFE
- The Glideins start Singularity and mount CVMFS



Fermilab

Singularity in GlideinWMS

- Singularity can be installed at the resource or used from CVMFS. Most resources support it.
- A VO can run using Singularity without extra effort
- Jobs can request a specific image (+REQUIRED_OS or +SingularityImage)
- The Glidein will invoke Singularity, start your image and mount the paths you need
- Jobs run in separate sessions (setsid()) to allow secure CVMFS

Singularity in GlideinWMS – more in detail

- Users and Resources negotiate the use of Singularity
  - GLIDEIN_Singularity_Use (FE) and GLIDEIN_SINGULARITY_REQUIRE (FA)
  - REQUIRED, PREFERRED, OPTIONAL or NEVER
- Define the images available to your jobs (key/ID, path or URL)
  - SINGULARITY_IMAGES_DICT (FE, FA)
- Select the image from the dictionary
  - REQUIRED_OS
- Mount your software and libraries
  - GLIDEIN_SINGULARITY_BINDPATH

Summary, questions

- Containers have enabled OSG and CMS to provide a safer and consistent environment across a large number of contributed compute resources, as well as provide a mechanisms for users to bring their own custom environments.
- Now we'd like FIFE Experiments to take advantage of them.

Resources

- More information:
  - https://opensciencegrid.org , https://display.opensciencegrid.org
  - http://wlcg.web.cern.ch
  - https://cernvm.cern.ch/portal/filesystem
  - https://support.opensciencegrid.org/support/solutions/articles/12000024676-docker-and-singularity-containers
    (short url: https://goo.gl/Yq9CYH)
  - http://glideinwms.fnal.gov/, http://glideinwms.fnal.gov/doc.prd/frontend/configuration.html#singularity

- Credits: Mats Rynge (OSG) and Dave Dykstra (Fermilab) provided most of the slides in this presentation. Thank you