# State of OSG Technology

Brian Bockelman
OSG AHM 2017

Simplify,
Simplify

# A Simpler, Kinder OSG

- Remember when OSG packaged its own MySQL?  Python?
  - OSG no longer re-packages system components.
  - Each release series, we remove about 25% of our software.  Year-over-year, this means impressive gains!
- A less complicated software stack increases the available effort for other OSG improvements!
  - This is especially true when retiring “orphan software” - that with a dead or “mostly dead” upstream where OSG takes on the support costs.

# Year of the Retirements!

- Somewhat unexpectedly, many of our (software) friends have been retiring en masse:
  - **GRAM**: Already gone from most sites for about a year.
  - **glexec**: To be replaced by a new component.
  - **GIP/BDII**: Replacement (OSG Collector) already integrated into HTCondor-CE.  You haven't used this in awhile.
  - **Gratia (central-only)**: Move from a monolithic MySQL database at FNAL to a decentralized architecture.  Database is ElasticSearch at Nebraska.
  - **bestman2**: Replaced by load-balanced GridFTP.
- Ideally, retiring components frees up your time to do other things!

# OSG, Then And Now

- About to start packaging with RPMs.
- Users submit jobs to the CE.
- CE based on Globus GRAM
- Info services based on GIP/ CEMon/BDII
- Storage Element model based on SRM.
- NFS for software distribution

- Experts at RPMs packaging.
- Users submit jobs to pilot-based systems.
- CE based on HTCondor-CE.
- Info services based on `condor_collector`.
- Active investigations to cache-based models.
- CVMFS for software.

# We aren’t done yet!

- I spent time reading the 2012 OSG proposal and am proud of how many of our original goals were achieved.

- I want to use my time today to outline how we’re trying to continue this theme of “simplification” over the the **next two years**.  Areas for improvement:
  - OSG CE
  - Runtime environment
  - The VO “zoo”
  - Authorization
  - Storage and data management
  - Monitoring

# Simplify the CE

- Retiring the GIP is a marginal decrease in total effort required to run a CE.
- Some sites are *retiring the CE itself:*
    - The BOSCO technology allows the OSG to host the CE on a VM run by OSG Operations.
        - The only site requirement is a password-less SSH connection to a site submit host.
- Great option for less-complex & new sites.
    - Delegates the *work* to the OSG but also delegates the *policy management*.
- In the meantime, we continue to chip away at any remaining rough edges in the HTCondor-CE.

# Simplify the Runtime Environment

- We want 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*.
- **Homogeneous / portable OS environments**: Make user OS environment as minimal and identical as possible.

Current approach?  Singularity

# What is Singularity?

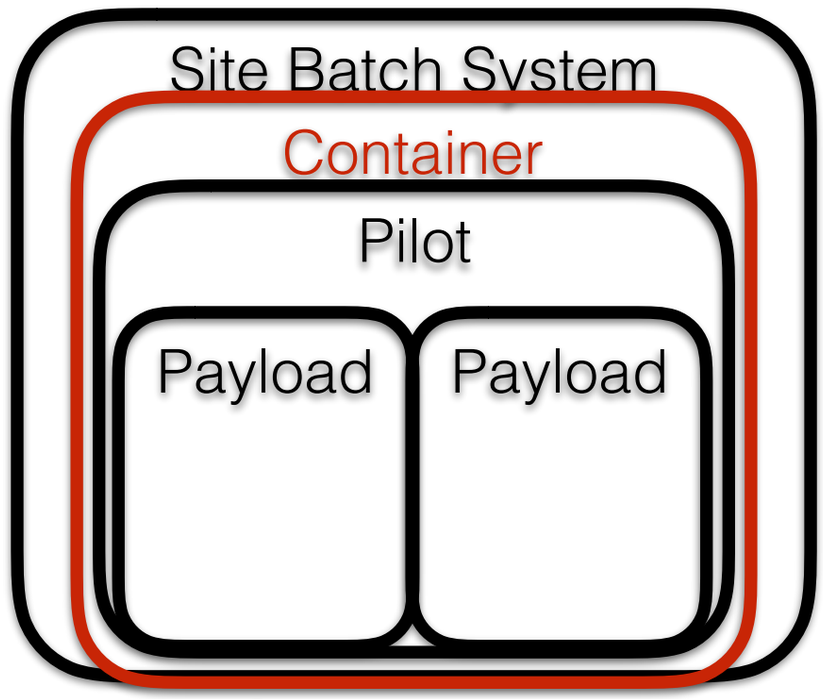
- Singularity is a container solution tailored for the HPC use case.
    - It allows for a portable of OS runtime environments.
    - It can provide isolation needed by CMS.
- Simple isolation: Singularity does not do resource management (i.e., limiting memory use), leaving that to the batch system.
- 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

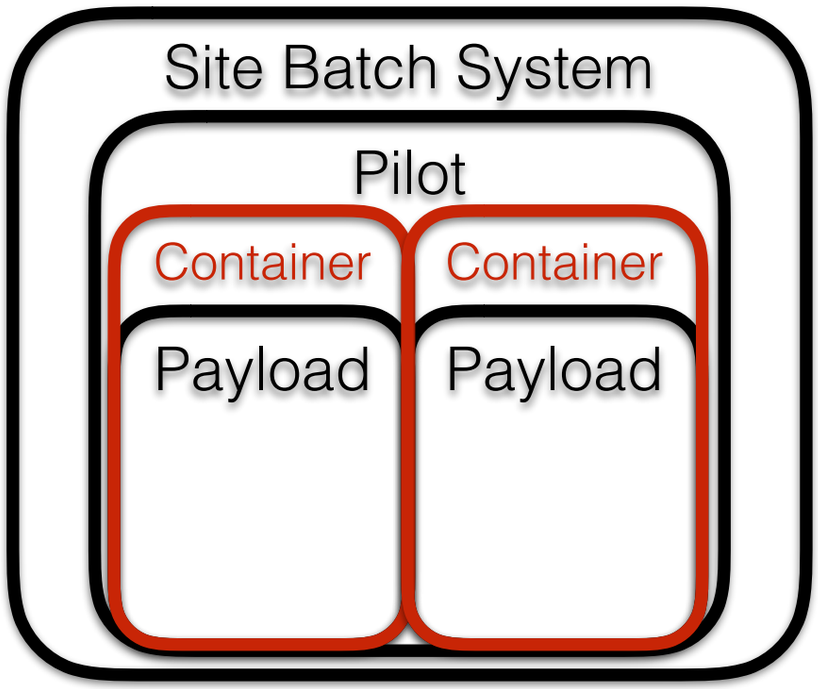
# **Who** is in a container?

- 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 Payload

```
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 System**

**Pilot**

**Singularity**

**User Payload only sees these processes**

# View From the Worker Node

User jobs are isolated from each other,
but it's still a familiar environment

**User Payload**
**only sees these processes**

```
\_ /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
```

# Simplifying the VO Zoo

- Setting up a classic VO is hard: **Why would you do that**?
    - *Policy enforcement*: **sites** can enforce policies specific to a VO; **VOs** can directly manage their share of resources.
    - *Isolation*: you do not want other VOs to interfere with your payloads.
- Singularity is one mechanism to provide isolation without needing a separate VO.
- In general, policy enforcement is *difficult*. However, we have tools for many simple policies!
    - Particularly, cases where site is “owned” by a single VO and everything else is opportunistic.
- The “support matrix” has (# VOs) * (# site) entries. Decreasing number of distinct VOs *as seen by the CE* saves effort overall. **Do you really need to submit your own pilots?**
    - Default GUMS template is **2,000 lines of XML** and **authorizes about 20,000 users** at the CE. We can do much better!

Looking forward to working with the community!

# OSG Authz Overhaul

- OSG is in the process of overhauling the authorization system. OSG's short-term goal is to replace `edg-mkgridmap`.

- Longer-term, OSG wants to drop GUMS.  Sticking point: pool accounts on storage.
    - Pool accounts on worker nodes are not needed once `glexec` is retired.

- I am in discussion with CMS security to determine whether pool accounts are actually needed.  Currently, they only are required for protections in `/store/temp/user`.
    - Assuming "no," one could retire GUMS this year.  More likely: 2018.

# OSG Authz Overhaul

- OSG authorization would primarily consist of two files:
  - `grid-mapfile`: site manual mapping of local user's DN.
  - `voms-mapfile`: mapping of VOMS FQAN to a local username.
    - OSG will ship a starter template.
- Site would be responsible to synchronize these files across services using Puppet/Chef/Ansible.
- There are a few other components (banning DNs), but things are still "synchronize a few simple policy files."

# (Web) Authentication Modernization

- Death to user certificates!  (*Well, in the browser*)

- OSG is working on upgrades to our web properties to eliminate the use of certificates to login.
    - Goal: login is done with your university ID.

    - OSG *does not* want to get in the business of maintaining usernames and passwords.

- “No user certificates” goal applies to sysadmins and OSG users.

- Likely approach: use **CILogon** to handle the authentication infrastructure pieces.

# Opportunistic “Storage”

- For about 8 years, OSG tried to make opportunistic storage (elements) happen:
    - Opportunistic computing is like filling empty seats on an airplane: it was going to fly regardless.
    - Opportunistic storage is like real estate: you don’t give it away!
- Indeed, **we lack the tools to allocate, manage, and utilize storage**.
- About two years ago, we shifted gears: focused on cache-based models instead of opportunistic storage.

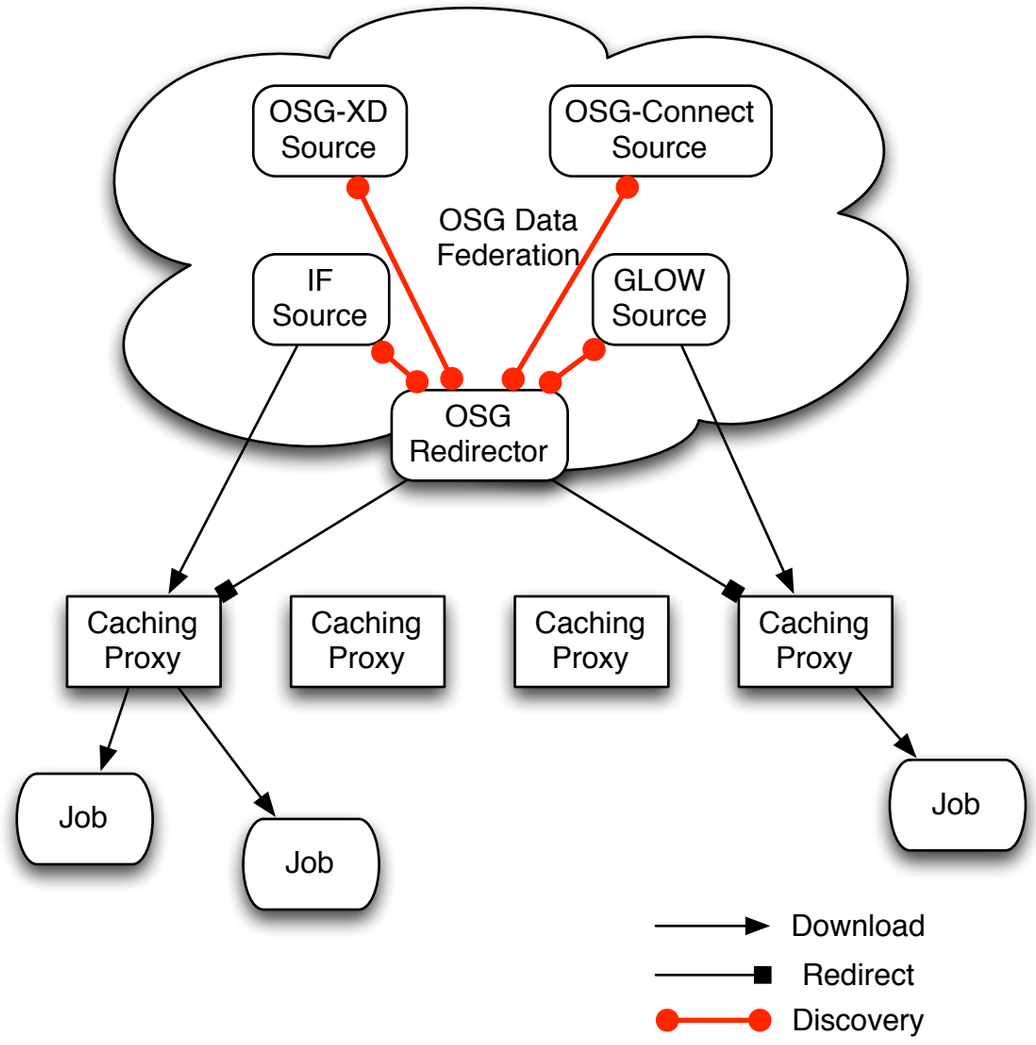
# It's all about storage management, stupid!

## Allocate B bytes for T time units

- **We do not have tools to manage storage allocations**
- **We do not have tools to schedule storage allocations**
- **We do not have protocols to request allocation of storage**
- **We do not have means to deal with "sorry no storage available now"**
- **We do not know how to manage, use and reclaim opportunistic storage**
- **We do not know how to budget for storage space**
- **We do not know how to schedule access to storage devices**

# StashCache in 2017

- StashCache is our data caching infrastructure. Relies on volunteer sites to run caching servers distributed .
    - Prevents VO from having to scale their own storage services *if* they have cache-friendly workflows.
- In 2016, we saw VOs use hundreds of terabytes a week.
- Multiple VOs use the StashCache infrastructure to distribute data through CVMFS.
- LIGO uses this to **securely** distribute their physics data.



# Future Topics: Monitoring

- Looking forward to *next year*, monitoring is in a sad state:
    - The existing tool, RSV, is OSG-specific.  Does not have many features found in popular monitoring solutions.  (It *does* have a few features unique to OSG!).
    - The probes often don't answer the question "Is my site working?"
- Approach future: **Embed monitoring** into the services themselves.
    - CE will report whether it is correctly configured.
    - GlideinWMS factory will report whether pilots are successful.
    - GridFTP (or XRootD) servers will self-test.
    - Continue to aggregate service data centrally — but **nothing for sites to run**!

# Areas Ignored

- Giving a status talk about OSG Technology is impossibly broad.  To deliver on the topic of *simplicity*, I skipped out on:
    - Maintenance of a broad range of “orphaned” software.
    - Slow and steady improvement of difficult services - Network Archive, OASIS.
    - Upcoming major HDFS upgrade this year.
    - *The fact we release software every month like clockwork*.
- These all take - or will take - immense effort and talent to pull off!
- Thank you to all the teams involved!

# Parting Shots

- Simplify, simplify, simplify: We are decreasing the “OSG footprint” at sites.

- For the most part, this has meant retiring components with duplicate or marginal functionality.

- In at least one case - Singularity - this means supporting a new piece of software.

- Innovation is disruptive!  I realize that, despite best efforts, “simplification” has often caused significant upfront work, sweat, and tears.

  - **THANK YOU** for remaining with us and contributing to the community!