The glideinWMS system: recent developments

Marco Mascheroni
University of California and San Diego

On behalf of the glideinWMS and the CMS Submission infrastructure teams

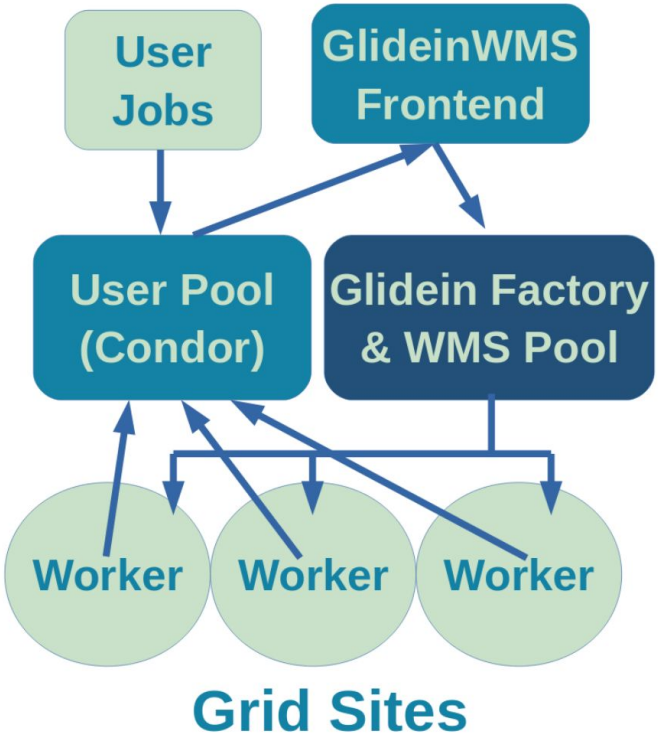
Outline

- GlideinWMS overview
  - With focus on is done on the worker nodes
- New features (Site related)
- Efficiency Improvements (CMS)
- Future works and Summary

GlideinWMS in a nutshell

Manages a dynamic condor pool that grows and shrinks based on user's request

- The glideinWMS frontend is the brain
  - It looks at the *user* jobs in the condor schedulers
  - Matches them over computing elements, aka *entries*
  - Determines number of *pilot* jobs to submit
- The glideinWMS factory is the arm
  - Submit the glidein_startup.sh script to entries (CEs) based on frontend requests
- Pilot jobs may run multiple user jobs
  - glidein_startup.sh configures and starts the condor startd daemon



What is done on the nodes (glidein_startup.sh)

- Making sure the resources we are acquiring are configured properly is extremely important. The pilot can be configured to execute validation scripts from both factories and frontends:
  - Factory for general validation
    - Check condor version, Collector setup, X509 proxy cert validation, Check if drain flag is active, publish statistics about the node if available
  - Frontend for experiment specific checks
    - Singularity validation, Cvmfs and OS checks, OS checks, Benchmark, Network and squid proxy setup, CPU and memory checks
- Connecting to the user pool
  - Once we know the nodes are ok, the glidein_startup.sh script writes out a condor_config file and starts the startd daemon

GlideinWMS deployment

- Two factories managing more than 400 entries
  - gfactory-1.t2.ucsd.edu for all the frontends
  - vocms0206.cern.ch is CMS exclusive
- Frontends (and average cores):
  - CMS => 160k (+T0 => 40k)
  - OSG (Connect)=> 30k
  - Glow and FIFE => 10k each
  - Gluex, LIGO, nanohub, Nova, LSST, IceCube => Few thousands

Only counting pilots as seen by the factory

New features (Site related)

Singularity support in glideinWMS

Singularity is becoming more and more popular, with some experiments making it mandatory.

- Different use cases to address. E.g.:
  - Experiments able to specify mandatory/optional/no use of singularity
  - Sites might want to force the use of singularity because of internal policies
- Support available since 3.4.2
  - Policies can be specified both on the frontend and the factory/entry level, and then they are combined

	NEVER	**OPTIONAL**	**PREFERRED**	**REQUIRED (REQUIRED_GWMS)**
NEVER	NEVER	NEVER	NEVER	FAIL
OPTIONAL	NEVER	NEVER	PREFERRED	REQUIRED
PREFERRED	NEVER	PREFERRED	PREFERRED	REQUIRED
REQUIRED	FAIL	REQUIRED	REQUIRED	REQUIRED
DISABLE_GWMS	DISABLE	DISABLE	DISABLE	DISABLE (FAIL)

Not only Grid

- Besides the usual suspects (Arc-ce, HTCondor-CE, cream) clouds are also supported
  - Support for EC2 and GCE is available.
  - Will start a specific VM image instead of sending a pilot to the CE
- Have been used to for CERN resources before condor was available
- Classic frontend not suitable for commercial clouds
  - Decisions only based on demand and available entries
- Evolution with the HEPCloud project
  - Decision engine replaces frontend
  - Many more policies can be added (e.g.: take into account cost)

Starting pilot jobs in a vacuum

- Allow sites that are not connected to any glideinWMS factory to join an existing user (glideinWMS) pool
  - Basically to allow them to add startds to a pool
- Still need to execute the glideinWMS frontend validation scripts (even if it is not needed for “pressure decisions”)
  - every time the frontend and the factory are reconfigured, the name of the downloaded script has to change to use squids
  - manual_glidein_startup allows to generate the glidein_startup.sh parameters on the fly

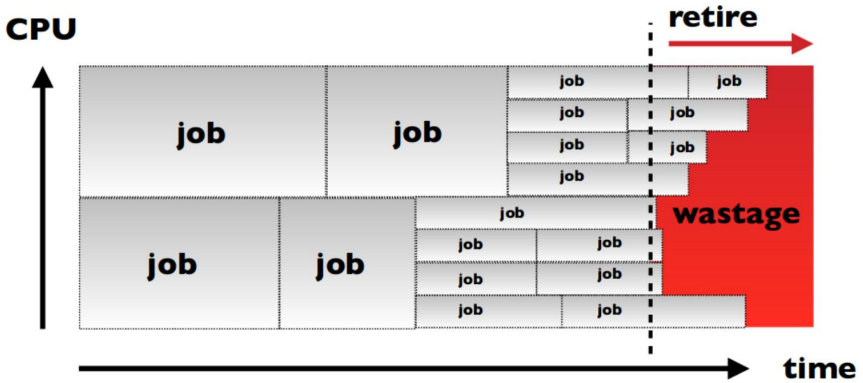
Automatic detection of worker node cpus/mem

- Usually the entry needs to be configured with number of CPUs and memory
- In case of worker nodes with different configurations more entries needs to be added to the factory
  - Operationally expensive!
- Possibility to configure entries to use the “wholenodes” feature added in 2018
  - Validation scripts will run and detect the available number of cores and memory, and then configure the condor startd accordingly
  - Frontend needs to know number of cores/memory per node to provide estimation
    - An estimation of the number of cores/memory still needs to be provided in the factory configuration

Efficiency improvements (CMS)

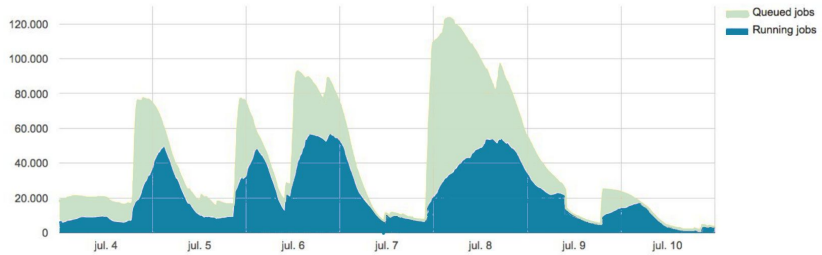
Fitting user's jobs into pilots

- Use of pilots offering many cores may cause fragmentation
  - And, in turn, wastages during retire time
- How can we use resources in a smarter Way?



- CMS uses resizable jobs for some workflows
  - Requested CPUs is an expression that changes once pilot slots are negotiated

Matching user demand and pilot pressure



- Poor scheduling efficiency because of bursty demand
- Pilots starting at sites after job pressure subsided

- Initial simple mechanism introduced in glideinWMS and adopted: regularly remove idle pilots
  - Not ideal since can cause unnecessary churn. Also experienced unexpected behaviors where jobs are not actually removed on some batch systems.
- More advanced feature to remove pilots based on frontend pressure available

Supporting local users

Give access to extra quota resources to local user using the CMS Remote Analysis builder (CRAB)

- Site admins can indicate the local user in different ways
  - Using voms roles
  - Listing users in an egroup
  - Listing users manually in siteconf
- Pilot with a special role will be requested by the frontend for a site if local jobs exist
  - The site admins can then route those pilots to the desired quejes
  - More info here

Automating factory configuration generation

- Currently factory configuration management is done manually
  - A lot human effort, lot back and forth with site admin
- Different information system could be leveraged to automate the process though
  - GOCDB, Topology in OSG, BDII in EGI
  - CMS will start adopting CRIC which already collects all those data!
- Prototype that uses CRIC to generate factory entry configurations has been implemented
- Milestone: have 20 automatically generated entries in production by July first

Summary

- New features have been introduced
    - Wholenodes, glideins in a vacuum, singularity support
- Efficiency has been a major area with many improvements implemented/introduced in 2018
- Automation of factory configuration is coming soon!
- Site defined policies is a topic of interest

Contact us at osg-gfactory-support@physics.ucsd.edu or via ggus tickets (Support Units: CMS Glidein Factory or CMS Submission Infrastructure)

Questions?