

# glideinWMS in the Cloud



**ANTHONY TIRADANI**

**AND THE GLIDEINWMS TEAM**

# glideinWMS

(review of basic principles)

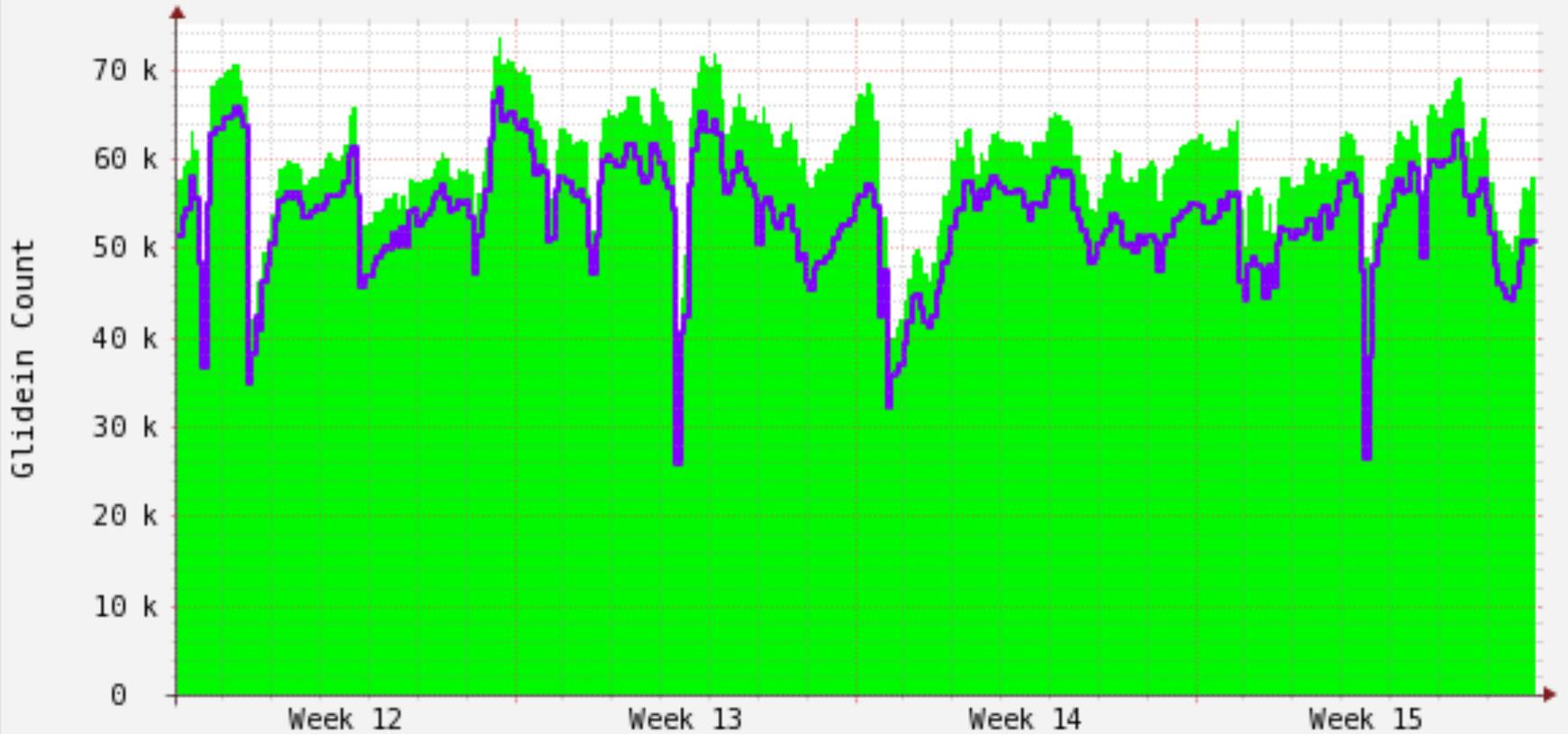


- Pilot-based WMS that creates an on demand dynamically-sized overlay condor batch system to address the complex needs of VOs in running application workflows
- Components
  - WMS Collector
  - Glidein Factory
  - User Pool Collector
  - User Scheduler
  - VO Frontend
- Factory knows about the sites and how to submit glideins to the sites
- VO frontend knows about the user job details
- WMS Collector acts as a dashboard for Factory - VO Frontend communication

# glideinWMS Worldwide (Grid)



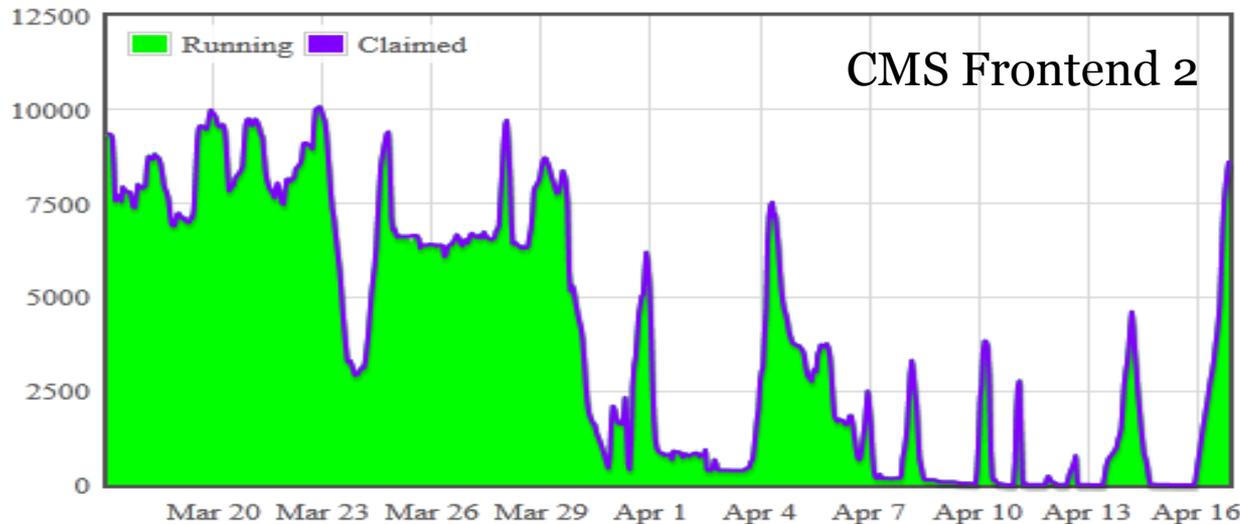
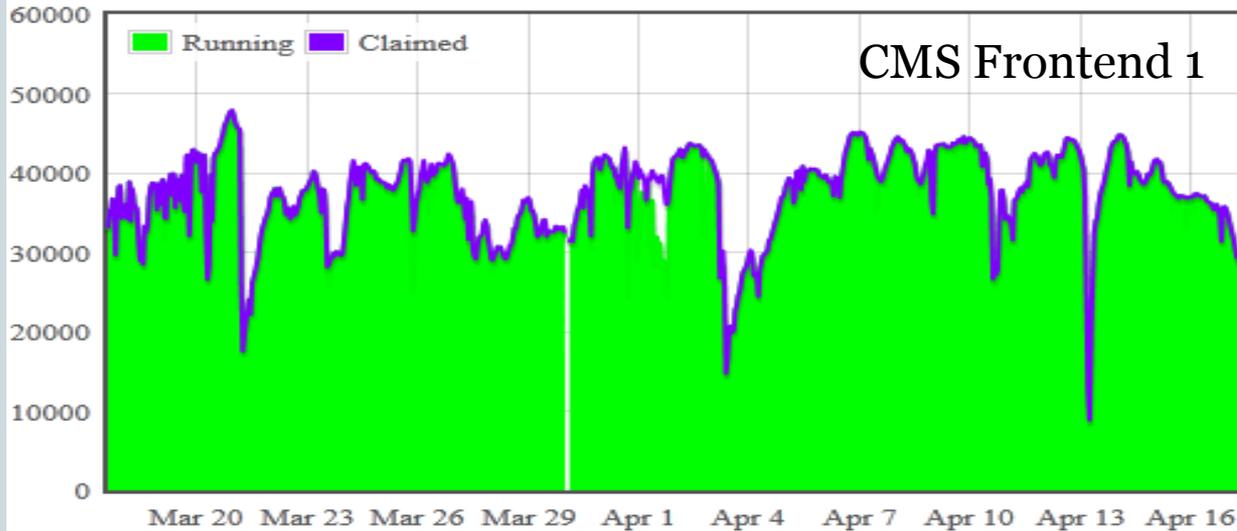
Aggregated Data for 4 Factories



Average Running Glideins: 59690

Average Claimed Glideins: 53538

# CMS glideinWMS Worldwide (Grid)



~60,000  
concurrent  
jobs!

# Cloud Definition

(for this presentation anyway)



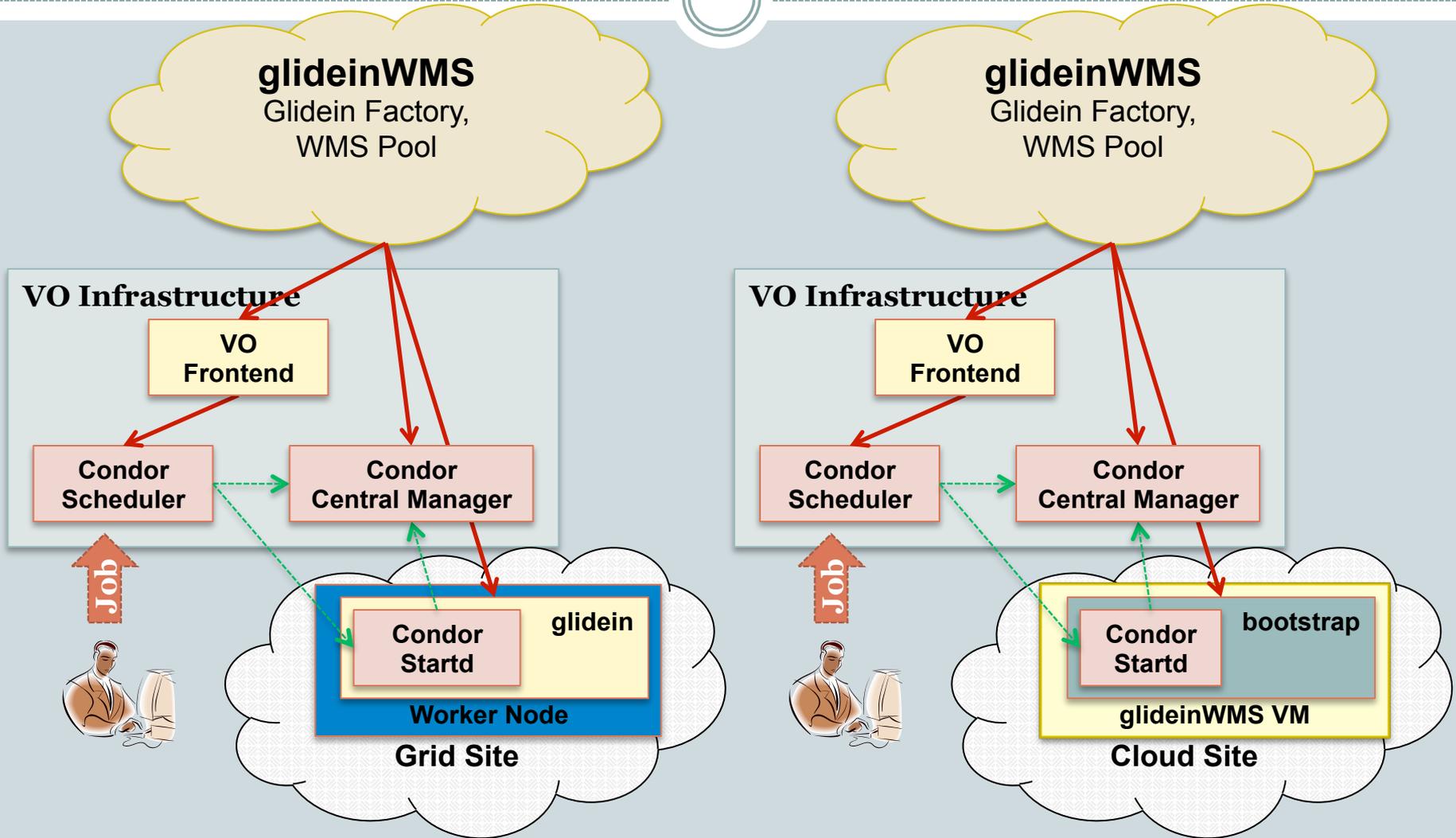
- **Cloud Definition:** Cloud in this presentation refers to the EC2 Infrastructure as a Service (IaaS) computing model
- **This does not limit the Cloud to Amazon's service**
  - Almost all alternative cloud service providers offer the EC2 Query API interface
  - Almost all alternative cloud software provides an EC2 Query API interface (e.g. Eucalyptus, OpenStack, ect.)

# Why Use the Cloud?



- Cloud offers choice of OS and system libs
  - Don't have to ask for specific versions to be installed on grid site worker nodes
- Can't get enough time on the Grid (deadline looming)
  - Get guaranteed resources in the cloud
  - Amazon will be happy to take your money
- You can have privileged user access to the VM
- You can have custom software installed for your jobs
- You do not need to set up an infrastructure for your resources (worker nodes)

# glideinWMS: Grid vs. Cloud



# Cloud Support in glideinWMS



- glideinWMS development version 3.0 has been released with support for EC2 universe submissions
  - <http://www.uscms.org/SoftwareComputing/Grid/WMS/glideinWMS/doc.prd/download.html>
- Requires a properly configured virtual machine image to be uploaded and registered with the target cloud infrastructure (more later if desired, time permitting)
- Cloud credentials now must be presented along with a valid grid/voms proxy
- A different economic model is in effect. For now, it is up to the VO administrators to monitor usage.
- The “Cloud pilot” is indistinguishable from a “Grid pilot” from the end user’s view point

# glideinWMS/Condor Decisions



- Several decisions were made along the way that impacted the current release.
  - Condor changed from using Amazon's SOAP interface to use the EC2 Query API
  - SSH Key management
  - User Data handling

# SOAP API



- Condor originally used the SOAP API
  - Worked fine for EC2 based cloud
  - Eucalyptus... Not so much.
    - ✦ Eucalyptus only supported very specific WSDL versions per Eucalyptus version
    - ✦ WSDL version support not much documented (unless you count code as documentation)
  - Amazon\_gahp had to be recompiled if new WSDL was needed or desired – too often
  - Required certificates for communication
    - ✦ This meant that the CA for each target cloud had to be available to Condor.
    - ✦ Not much of a problem for Amazon, but can be difficult for more obscure clouds
    - ✦ Also can be difficult depending on how Condor was installed and how CA's are managed for the site

# EC2 Query API



- **Condor switched to EC2 Query API**
  - Nimbus, OpenStack, and others were coming out with support for the EC2 Query API
  - Eucalyptus works with the EC2 Query API
  - Becoming the standard Cloud API for all vendors
  - Only have to recompile `ec2_gahp` when support for a new method is desired
  - Does not require certificates – uses an Access Key and a Secret Key
  - Has worked quite well for us

# SSH Key Management



- The glideinWMS Factory serves multiple communities
- glideinWMS does not perform file management for job submission
  - e.g. no file spooling
- Condor creates an ssh key for each vm request submitted.
  - Past reasons for doing this include idempotency on Amazon EC2. (no longer necessary)
  - However, it is very convenient for multiple community support
  - Should a key be compromised, only one VM is compromised

# User Data Handling



- **Condor JDL snippet:**

- `ec2_ami_id=$ENV(AMI_ID)`

- `ec2_instance_type=$ENV(INSTANCE_TYPE)`

- `ec2_access_key_id=$ENV(ACCESS_KEY_FILE)`

- `ec2_secret_access_key=$ENV(SECRET_KEY_FILE)`

- `ec2_keypair_file=$ENV(CREDENTIAL_DIR)/ssh_key_pair.$(Cluster).$(Process).pem`

- `ec2_user_data=$ENV(USER_DATA)#### -cluster $(Cluster) -subcluster $(Process)`

- To avoid having to write out a JDL per submission, almost all dynamic data is passed as environment variables
- The “tricky” part was how to pass the Cluster and Process to the pilot. (Used to help debug problems)

# Future Work



- **Set new pilot submission limit:**
  - limit by budget
  - Provide VO Frontend directive: “Kill all cloud pilots, we are going to go over budget soon”
- **Improve accounting for cloud pilots**
- **Retrieve logs from “stuck” cloud pilots**
  - Might include a new feature request to Condor
- **Bullet-proof virtual machine management**
  - Condor will start and stop the virtual machines reliably, but what happens if the pilot stops working?
- **Release an RPM for configuring the Cloud image for glideinWMS**

# Questions



- Questions?

# Acknowledgments



- glideinWMS is an open-source Fermilab project in collaboration with UC-San Diego, the Information Sciences Institute, UW-Madison and the Condor team, and many others. We welcome any future contributors: [github.com/holzman/glideinWMS](https://github.com/holzman/glideinWMS)
- Fermilab is operated by Fermi Research Alliance, LLC under Contract No. DE-ACos-07CH11359 with United States Department of Energy.
- The US National Science Foundation under Grants No. OCI-0943725 (STCI) and PHY-0612805 (CMS Maintenance & Operations)
- the US Department of Energy under Grant No. DE-FC02-06ER41436 subcontract No. 647F290 (OSG)

# Extra Slide - Cloud Image Challenges



- **Who “owns” image?**
  - Who creates it?
  - Who patches it?
  - Who gets called when something goes wrong?
- **Where does the image reside?**
  - It must be pre-staged to the cloud infrastructure
  - Under which account is the image stored?
- **How do you create the image?**
  - What tools do you use to create and deploy the image?
  - How do you keep track of the image? (Image Catalog)

# Extra Slide - Cloud Image Challenges (cont.)



- What “extra” resources does the cloud provide?
  - How much memory is allowed?
  - How much “instance storage” is given?
    - ✦ This is a real issue for CMS
- How do you debug problems with your image?
  - With out knowing an admin for the cloud, or having a working example image, it can be frustrating to build a working image