Fermilab

Managed by Fermi Research Alliance, LLC for the U.S. Department of Energy Office of Science

# Cloud Services for the Fermilab Scientific Stakeholders

Steven Timm[1], Gabriele Garzoglio[1], Parag Mhashilkar[1], Joe Boyd[1], Gerard Bernabeu[1], Neha Sharma[1], Nicholas Peregonow[1], Seo-young Noh[2], Hyun Woo Kim[1], Sandeep Palur[3], Ioan Raicu[3]

Fermilab[1], KISTI[2], Illinois Institute of Technology[3]

# Outline

- Introduction—Why run workflows on Cloud
- Large Scale Demo on Public and Private Cloud
  - Unified Submission and Provisioning
  - Code Caching
  - On-demand Launch of Services in FermiCloud
  - Image Conversion Script
  - Results of Simulations
- Current Work and Future Plans
- Summary and Conclusions
- Acknowledgements

# Why Run Experimental Workflows on Cloud

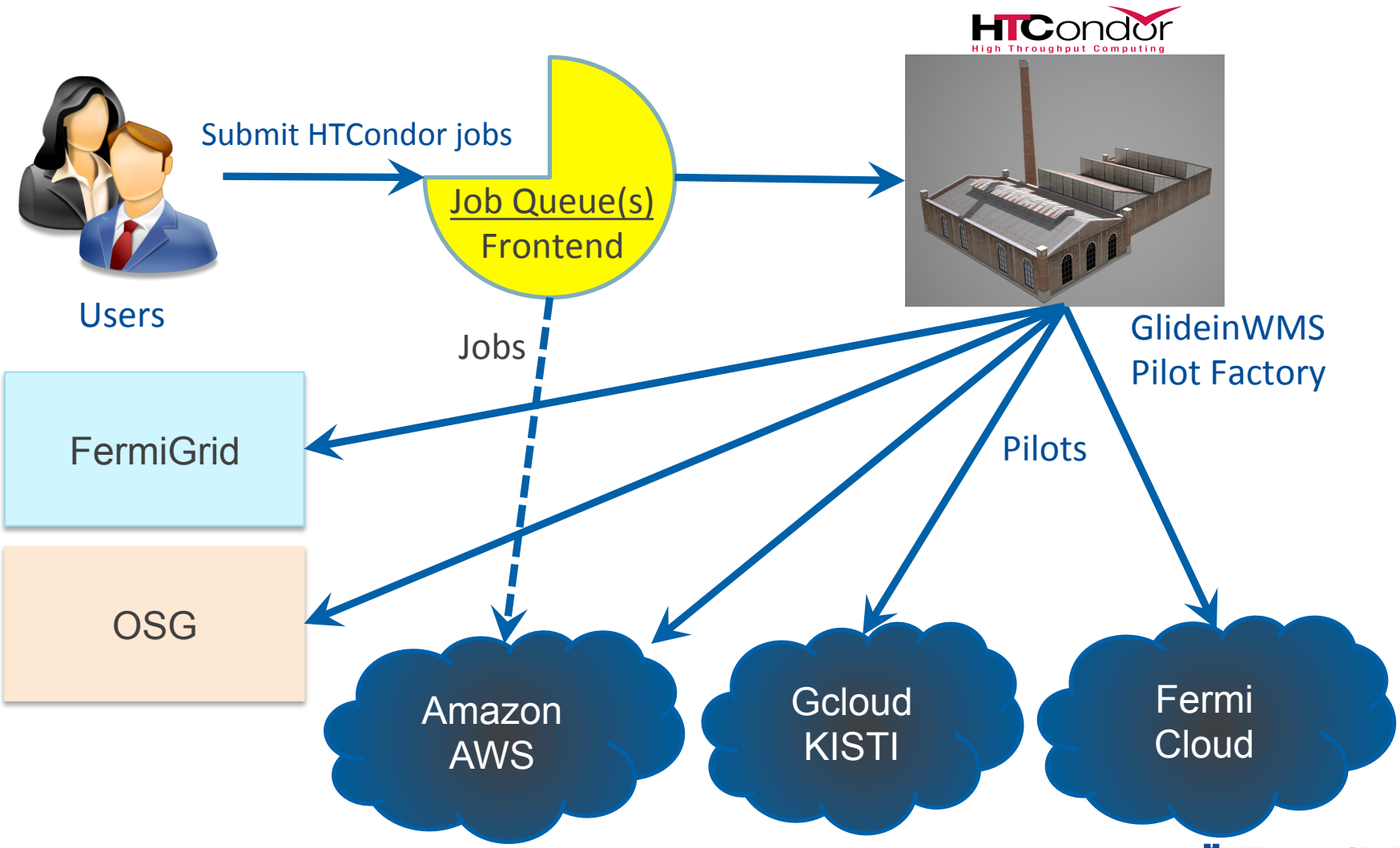
- High Energy Physics compute loads are highly variable
  - There is a constant DC load with peaks just before conferences
  - Also unusual job requirements—high memory, extra disk
- Fermilab has multi-year program of work to make software more portable to run on off-site grids and clouds
  - NOvA experiment has led the way
    - **Details in poster:** Large Scale Monte Carlo Simulation of neutrino interactions using the Open Science Grid and Commercial Clouds (Session A 465)
- Goal is to use commercial clouds to cover the peak demand
  - If Grid cycles are busy at Fermilab, likely busy everywhere
  - Identify shortcomings running on cloud and devise solutions (auxiliary services) as needed
- This R+D work done in summer/fall of 2014 as part of joint Fermilab/KISTI collaboration
- Fermilab now working to make commercial clouds an integral and transparent part of our Facility

# Definitions:

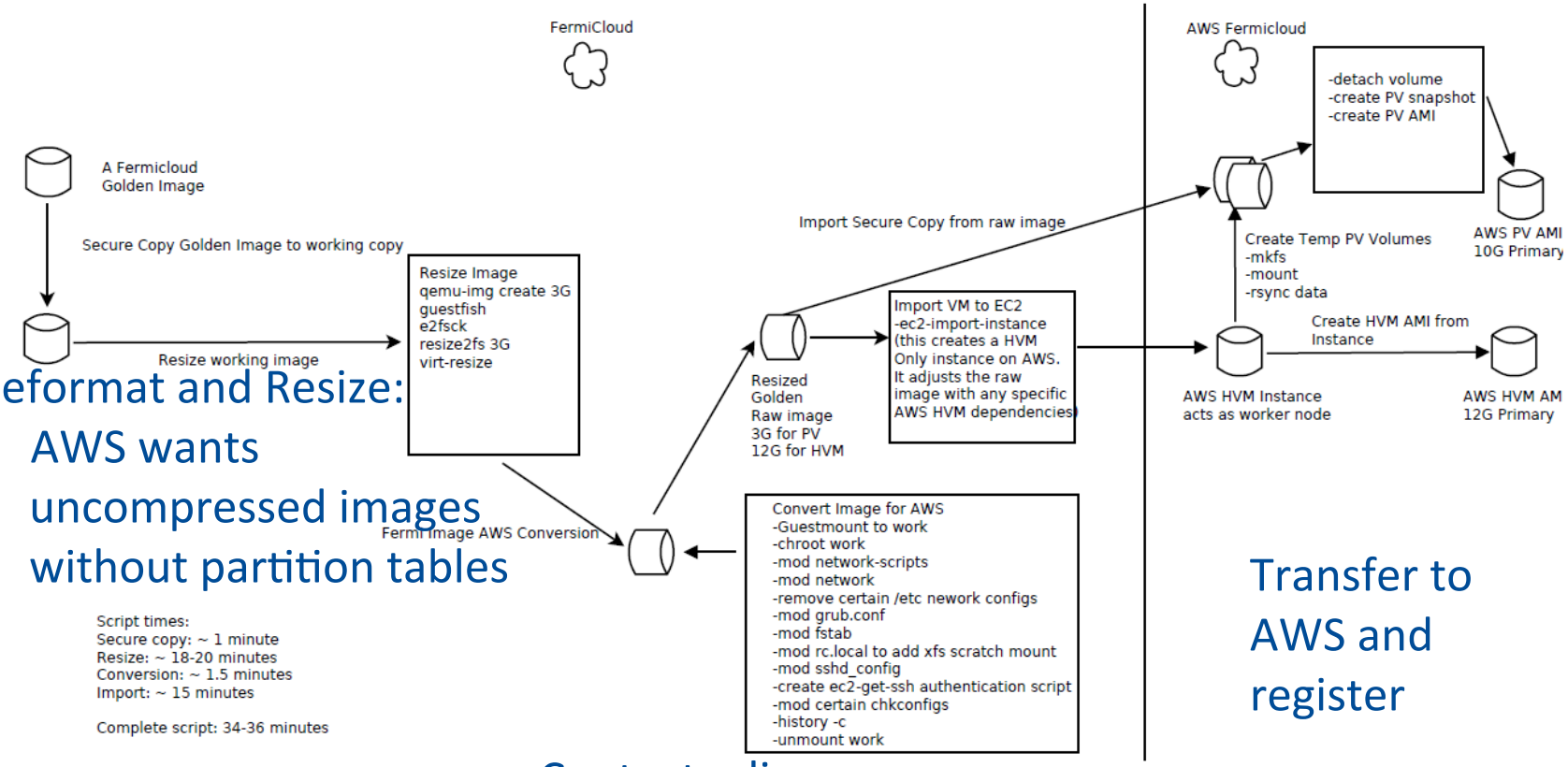
- The Facility
  - All computing and storage both on-site and off-site that is provided for the use of the experiments.
  - Virtual Machine launching service to obtain cloud resources
  - Dynamically scalable service instantiation
  - Provides auxiliary services to support the batch slots
    - Code Caching
    - Data Caching
    - Batch Submission

- Provisioning
  - The process of contacting a local or remote grid or cloud to acquire a “slot” for use in batch computing

# Provisioning via GlideinWMS – Grid and Cloud Bursting



# FermiCloud to AWS Image Conversion Tool




Reformat and Resize:

- AWS wants uncompressed images without partition tables

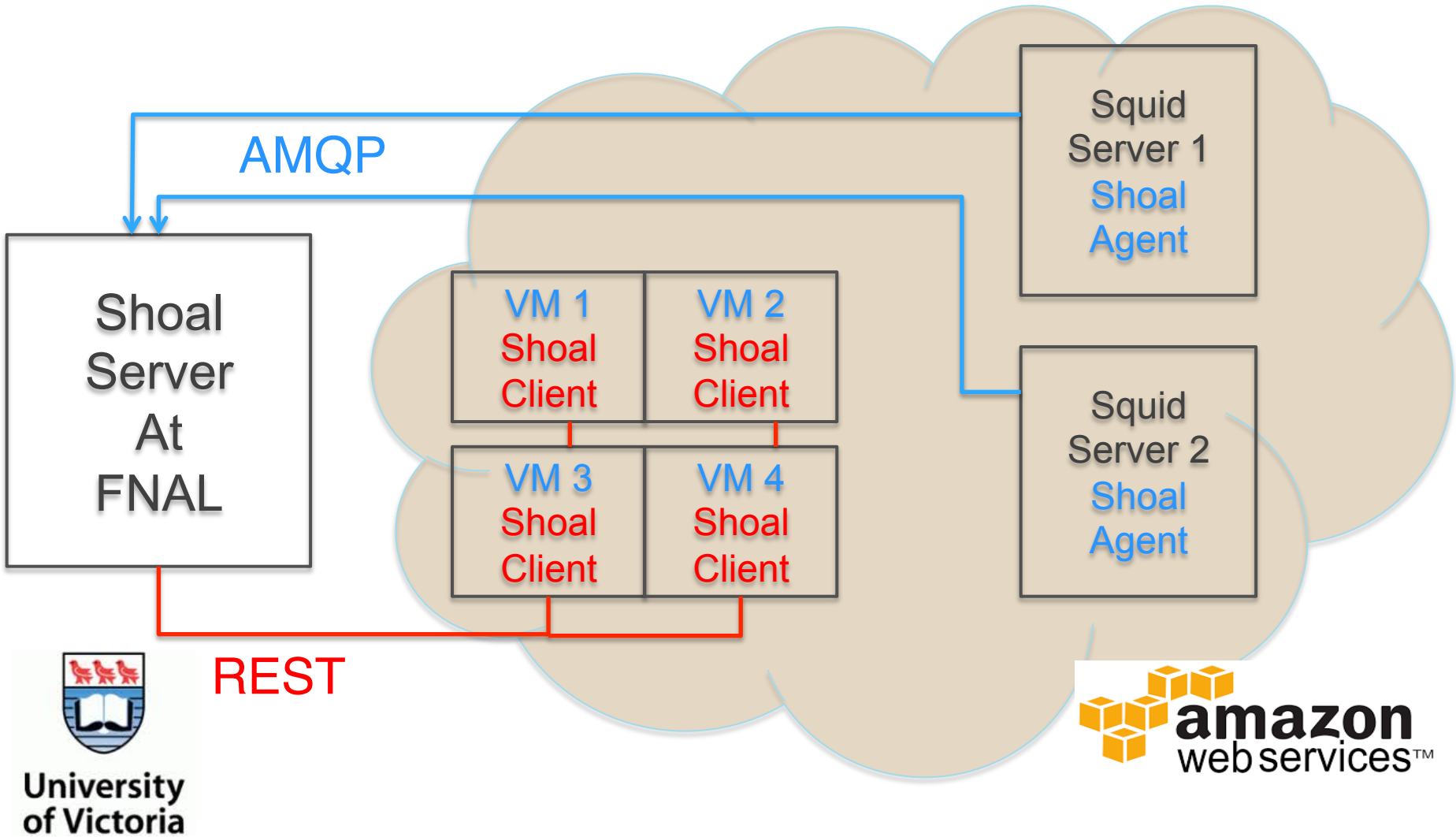
Script times:
Secure copy: ~ 1 minute
Resize: ~ 18-20 minutes
Conversion: ~ 1.5 minutes
Import: ~ 15 minutes

Complete script: 34-36 minutes

Contextualize:
Remove Fermi-specific, Add AWS-specific

Transfer to AWS and register

# SHOAL squid discovery system



# Results—NOvA Near Detector Cosmic Ray Simulation

- Main goal in 2014 was to show we could run physics applications of the Fermilab neutrino program on the cloud (public and private) at 1000-VM scale
  - Unlike their LHC counterparts, these experiments not used to running in distributed mode.
  - Educate them about the heterogeneity of external resources
  - Removed NFS dependencies from code and workflow
  - Code distributed via CVMFS—needed squid on cloud
- Results
  - 20,000 physics jobs total run in several trials
  - 3300 simultaneous jobs on AWS and FermiCloud in biggest trial
    - Up to 1000 each simultaneously on AWS and FermiCloud
  - Results sent back to Fermilab dCache servers in the FTS "Drop box"—largest set generated 467GB of Output

# 1000-VM Test: Amazon AWS Setup

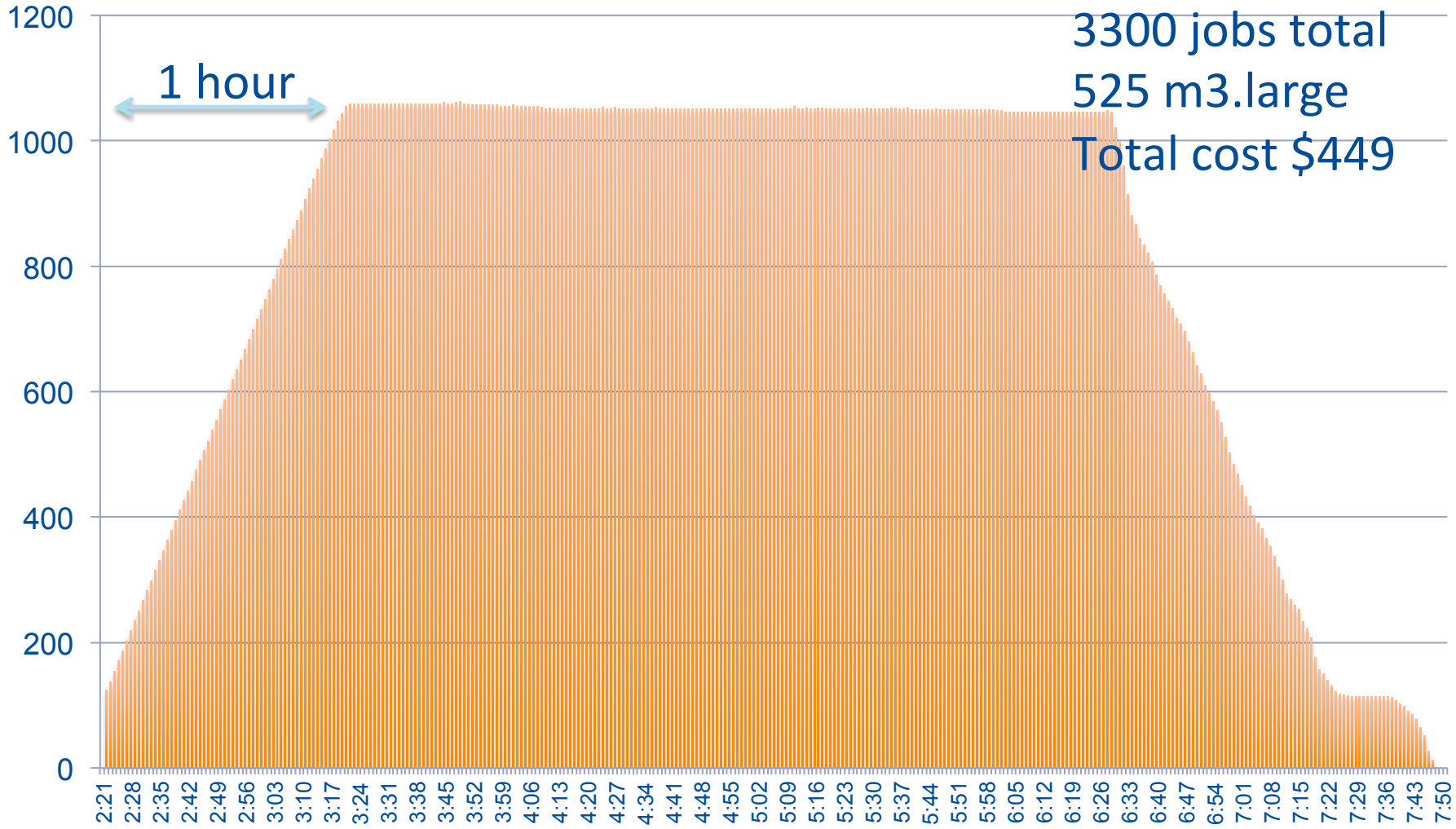
- Used m3.large instance which can run 2 jobs at once.
  - 2 cores, 7.5GB RAM, 30GB SSD Disk

- Costing (Total: ~ $450)
  - Data Transfer Charges: $51 of data transfer charges
    - Much less due to shorter log files
    - Amazon will give ESNet, Internet2 abatement on data egress charges
  - Compute charges: $398
    - 525 VMs @ $0.14 per VM/Hour
    - We can save a factor of 5-10 with spot pricing

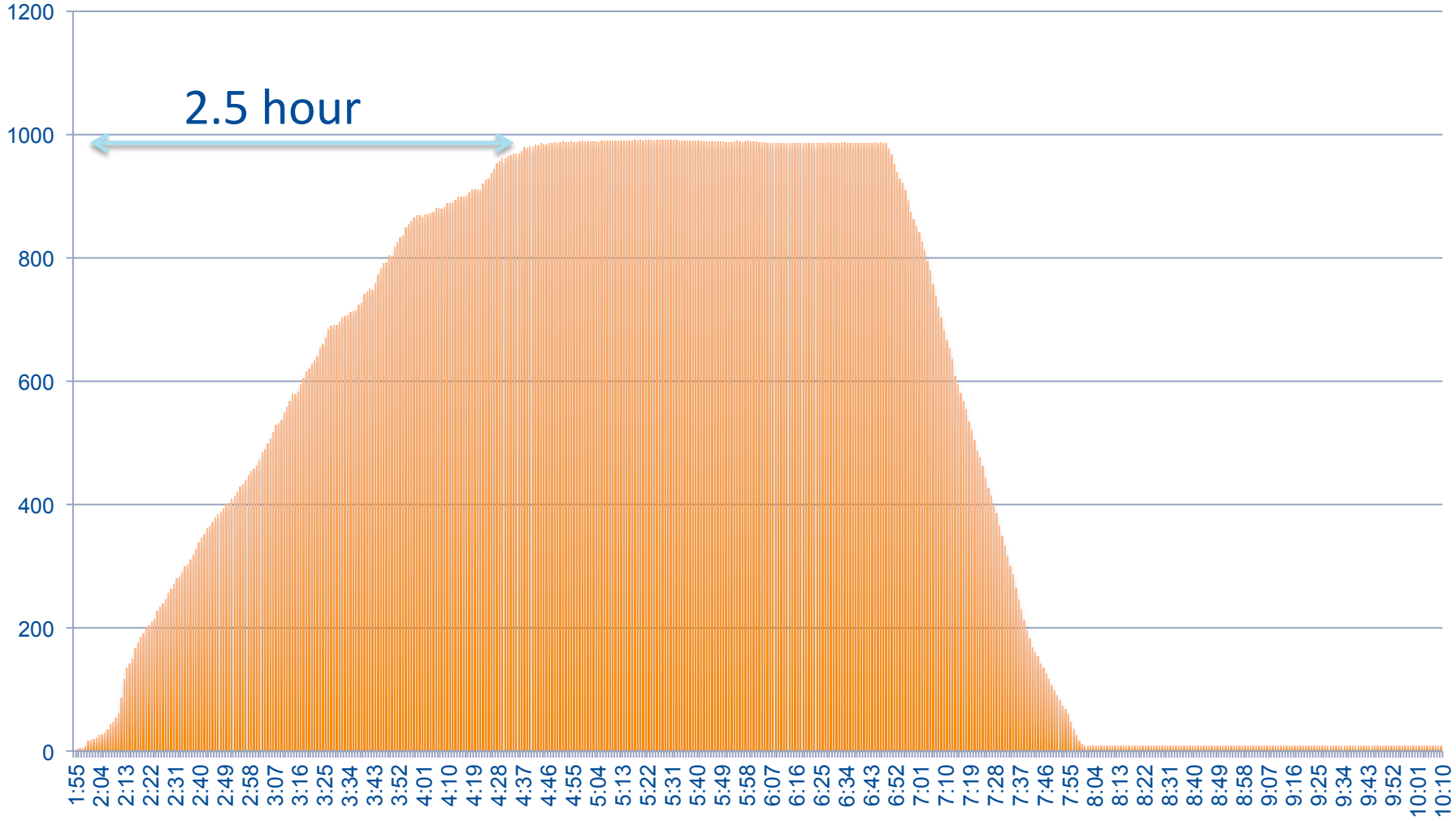
# 1000-VM Test: FermiCloud Setup

- OpenNebula 4.8 “econe-server” with X.509 authentication
- Routable private network—Routable inside Fermilab, NAT to outside.
- OpenNebula Host Machines
  - 140 Dell Poweredge 1950 servers, formerly part of CDF Farms (vintage 2007) 8 cores, 16GB RAM.
- Bluearc NFS as image datastore
  - Qcow2 image is copied to each node and run from local disk.
- OpenNebula’s own CLI could launch 1000 VM’s easily in about 30 minutes
- Issues (all have been worked around temporarily and reported to OpenNebula and HTCondor developers):
  - HTCondor use of OpenNebula API creates one ssh keypair per VM, total number of allowed keypairs is 300
  - Database growth sometimes causes the DescribeInstances call to time out. Can be worked around by aggressive pruning of database.

# Running AWS Jobs as function of time, Oct 23. 2014



# FermiCloud jobs as a function of time, Oct 23, 2014

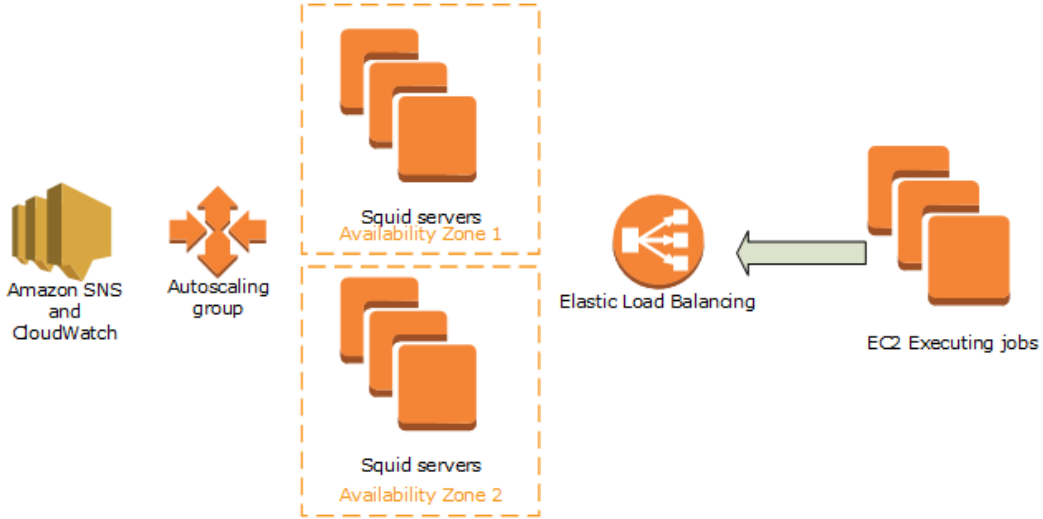


# Extending Facility Into Public Cloud (2015-2016)

- Physics Goals:
  - New: Run the most data-intensive NOvA applications on AWS
  - Event Reconstruction—six campaigns
    - 10000 input files, 250MB input, 250 MB output, 3hrs/job
  - Beam Simulation with Flux Files—ten campaigns
    - 38000 input files, 5 hrs/job
  - Will add 3-4 other use cases from other experiments
- Sustainability Goals:
  - Total expected AWS usage 2.1 million hours (100 x 2014 test)
  - For scale, NOvA alone ran 10.2 million hours in 2014
  - 145 million hours/year available on non-CMS FermiGrid.

# Code & Data caching: AWS Squid Service On-Demand

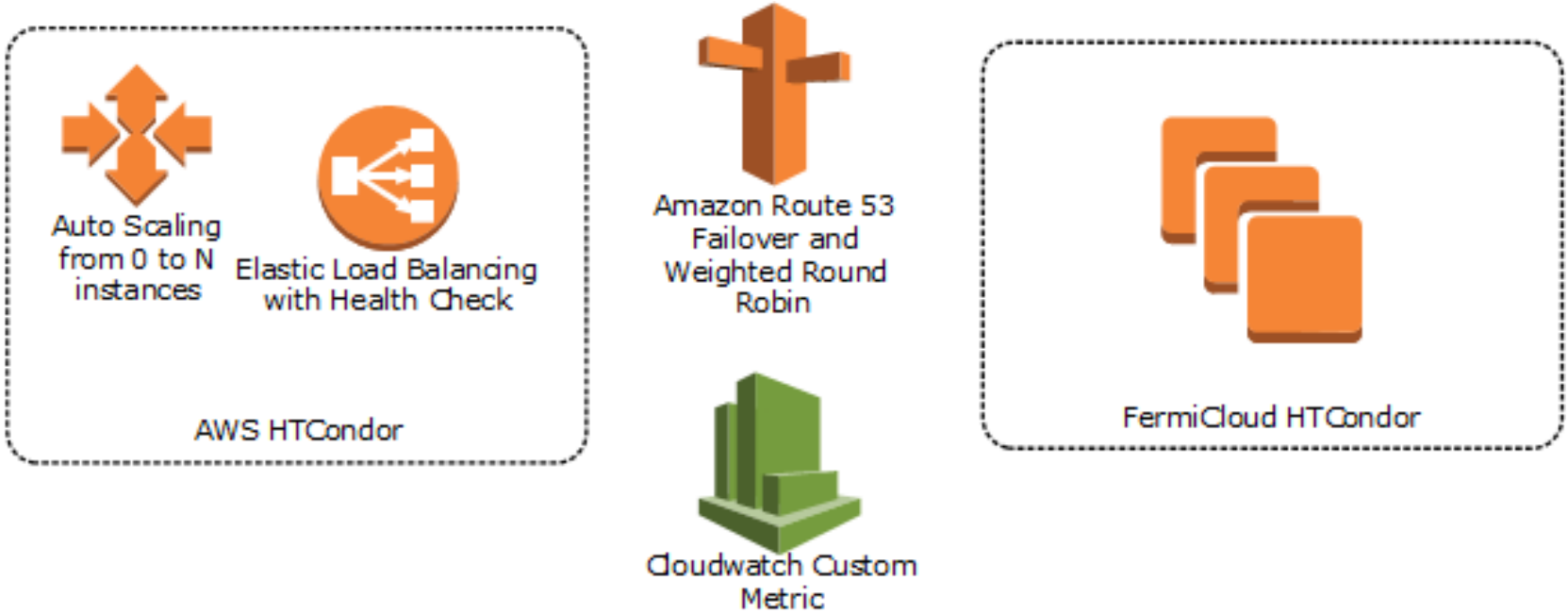
- Working Proto-type
  - Using AWS Autoscaling group feature instead of SHOAL (Slide 7) for scalable code and data transfer to AWS VMs
- Squid
  - A stateless service
  - Easy to scale up and down.
- Auto-scalable squid servers using AWS Autoscaling group
  - Deploying and destroying in 30 seconds using CloudFormation script
    - Autoscaling group requires at least one VM up all the time
    - Incurs charges even when the service is not required
    - CloudFormation script gets around the limitation by starting the Autoscaling group as required



# Hybrid Cloud: FermiCloud & Amazon Web Service

- Use FermiCloud first & Pay AWS only during spikes
  - Need to make differences in cloud providers transparent
    - User jobs don't notice any difference
    - Solution should be manageable by the operations

Have prototyped using Amazon Route 53 service to have a single service IP



# Summary and Discussion

- Technology discussed here is not new:
  - Virtual Machines—1971
  - HTCondor—1988
  - GlideinWMS—2003
  - Amazon Web Services—2006
- Emphasis is new:
  - Make the policy and architecture to integrate cloud into facility
  - Use the cloud like a cloud, not extension of the grid
  - Active strategic partnership with commercial cloud vendors
  - Focus on reliability and sustainable operations
  - Do hardest data-intensive computing on cloud
  - Unify grid and cloud provisioning in the facility, saving efforts so several VO's don't each have to do their own

# Acknowledgements

amazon web services™

OpenNebula

Scientific Linux

KiSTi
www.kisti.re.kr

ILLINOIS INSTITUTE OF TECHNOLOGY

ISSNAF
Italian Scientists and Scholars of North America Foundation

University of Victoria

NOvA

HTCondor
High Throughput Computing