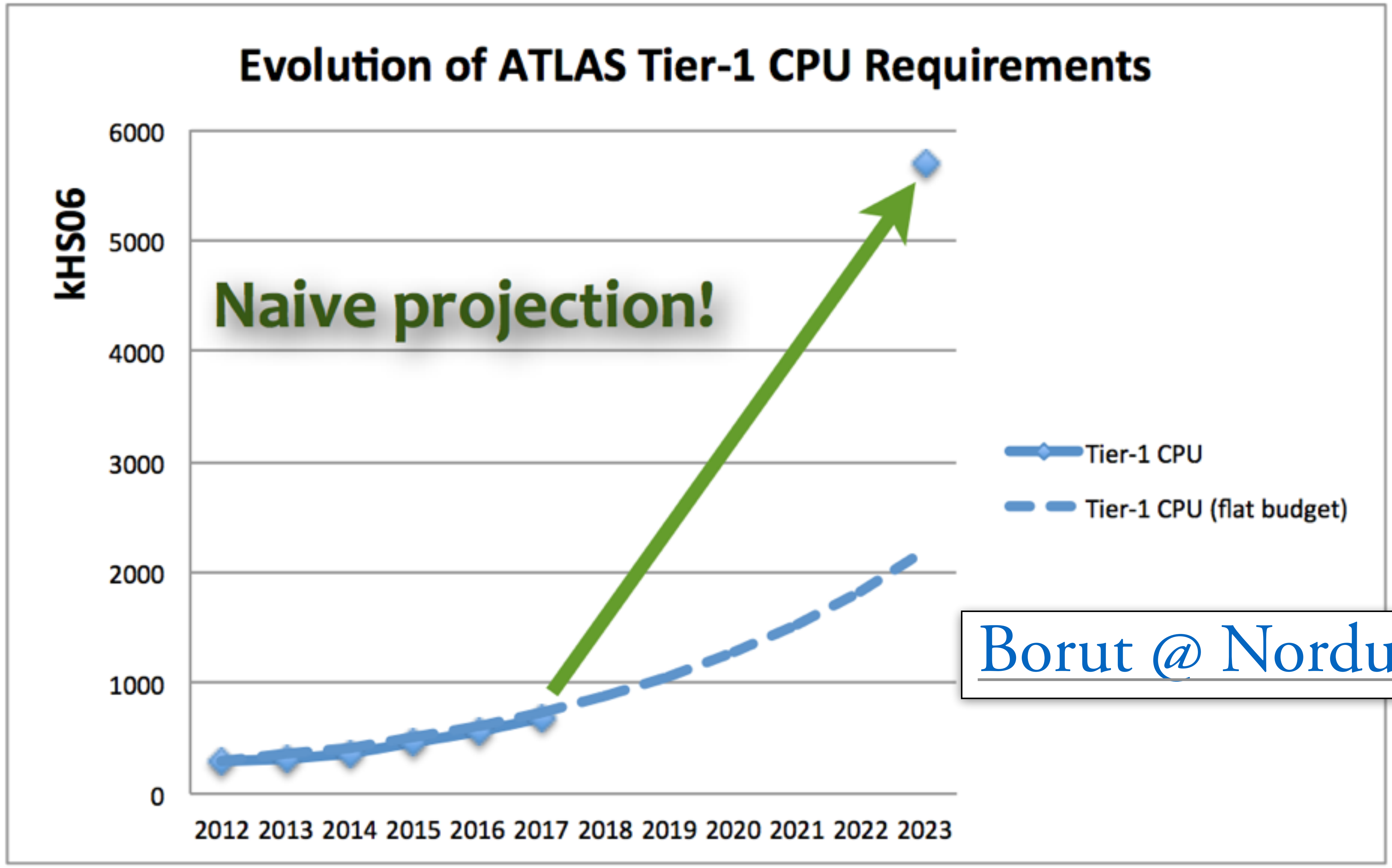
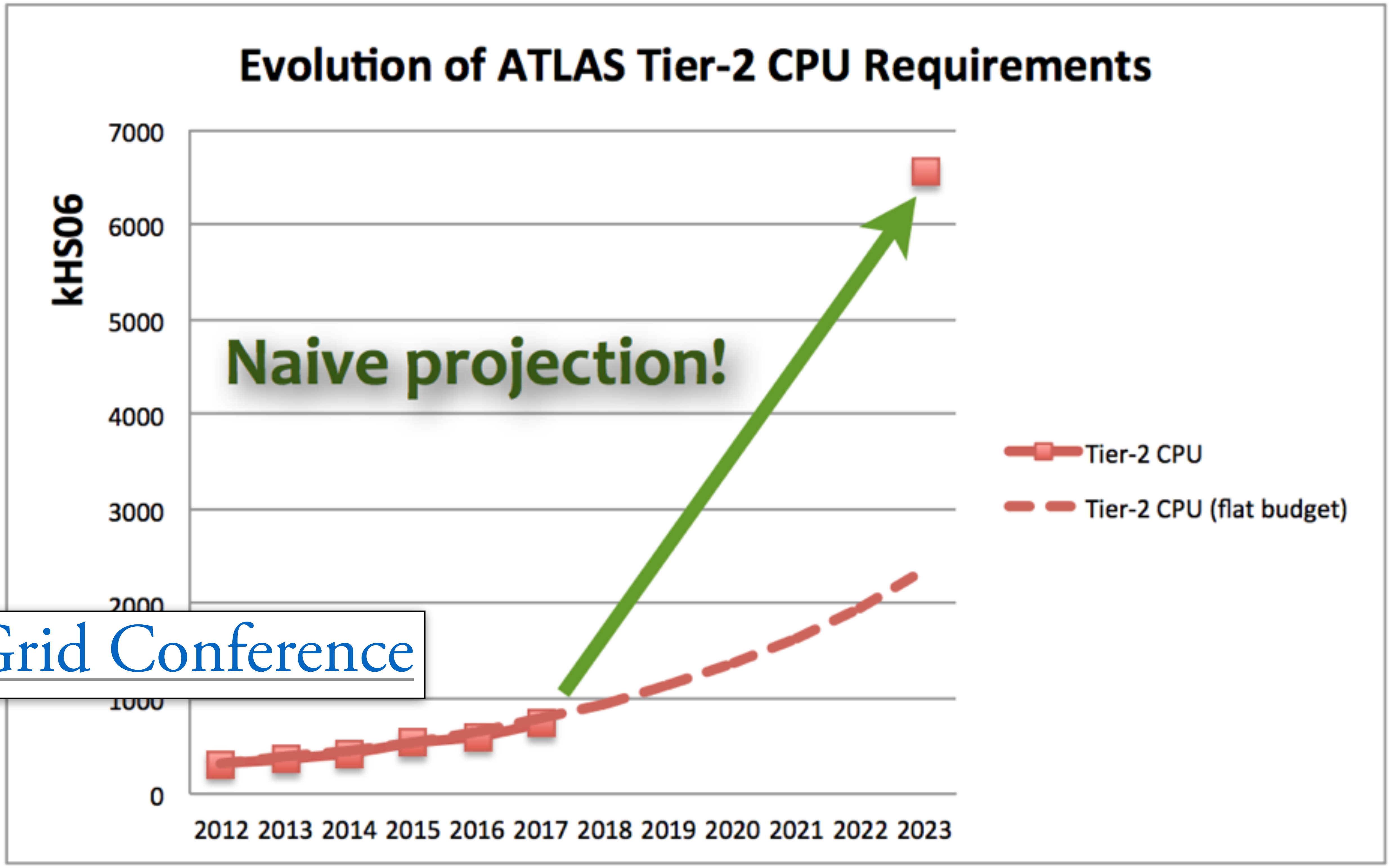
Developments in Architectures and Services for using High Performance Computing in Energy Frontier Experiments

Speakers: J. Taylor Childers (Argonne) and Lisa Gerhardt (NERSC)

Some of this work sponsored by the DOE HEP Center for Computational Excellence

LHC Grid Computing


Evolution of ATLAS Tier-1 CPU Requirements


Evolution of ATLAS Tier-2 CPU Requirements

Borut @ NorduGrid Conference

- Projected Computing needs for the High Luminosity LHC vary, but it is clear going from processing 30fb^{-1} per year to 300fb^{-1} per year will not keep up with flat budgets.
- HPCs will be an important tool for meeting the needs of the HL-LHC.
- In the US, ATLAS is already using >100M CPU-hours per year on HPCs.

High Performance Computers as a Bridge to HL-LHC

- 48k Nodes: 64 threads, 16GB each
- 1.6 GHz BlueGeneQ PowerPC
- 3.1M parallel threads possible
- 6.8B core-hours/year (Grid ~2.5B/year)

- 9,304 nodes: 68 cores x 4 HW threads (272 threads/node)
- Intel Xeon Phi (Knights Landing)
- 16GB on-chip memory
- 96 GB DDR4 2133 MHz
- 5.5B core-hours/year (Grid ~2.5B/year)

OLCF
Oak Ridge Leadership Computing Facility

- 18,688 nodes: 16 CPU cores, 1 NVIDIA Kepler GPU
- 2.2GHz AMD Opteron with 32GB
- 6GB RAM on GPU
- 2.6B CPU-core-hours/year

High Performance Computers as a Bridge to HL-LHC

- 48k Nodes: 64 threads, 16GB each
- 1.6 GHz BlueGeneQ PowerPC
- 3.1M parallel thre
- 6.8B core-hours/y

- 9,304 nodes: 68 cores x 4 HW threads (272 threads/node)
- Intel Xeon Phi (Knights Landing)
- 16GB on-chip memory
- ar)

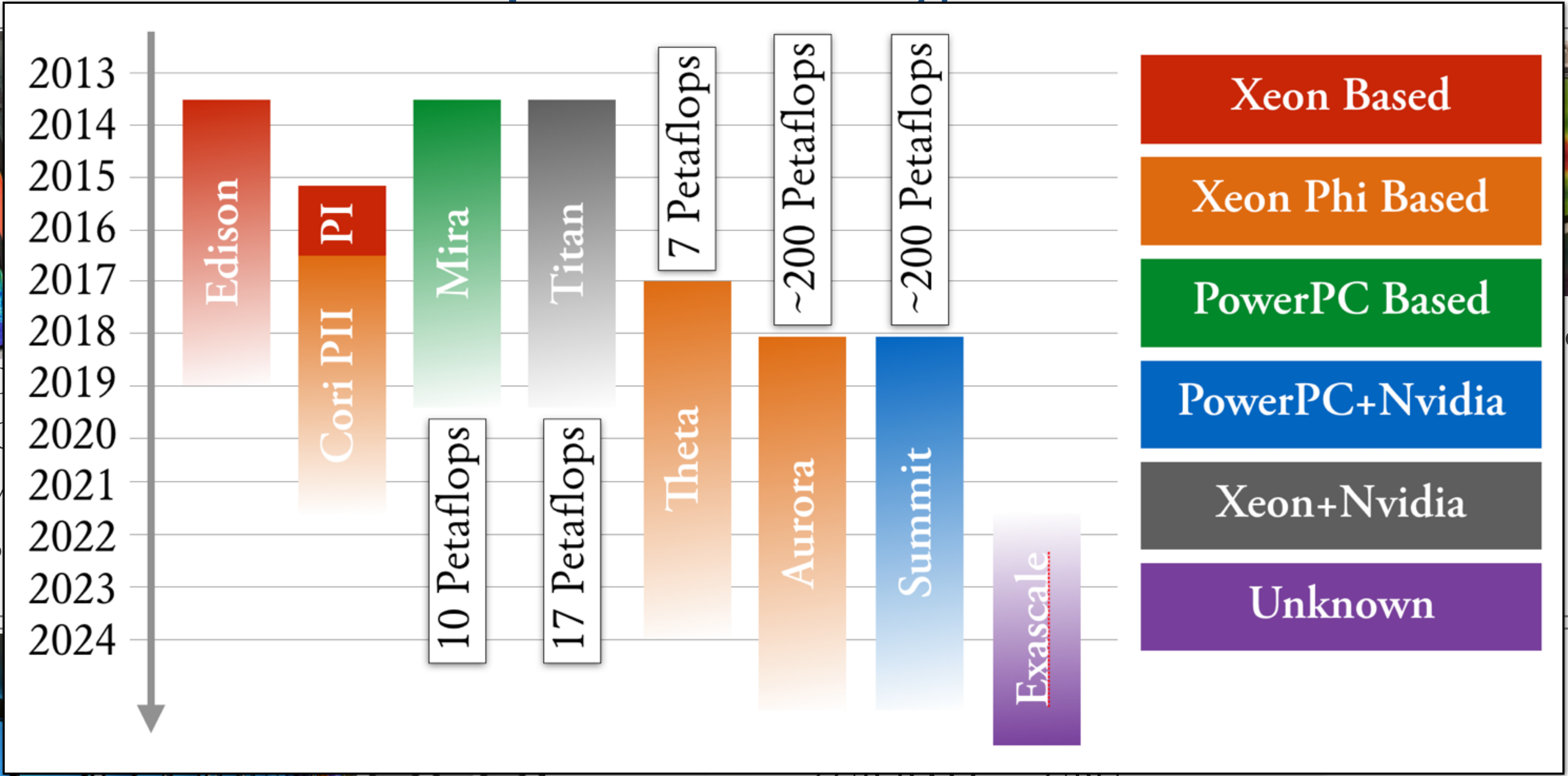
Currently: 15B core-hours per year on HPCs

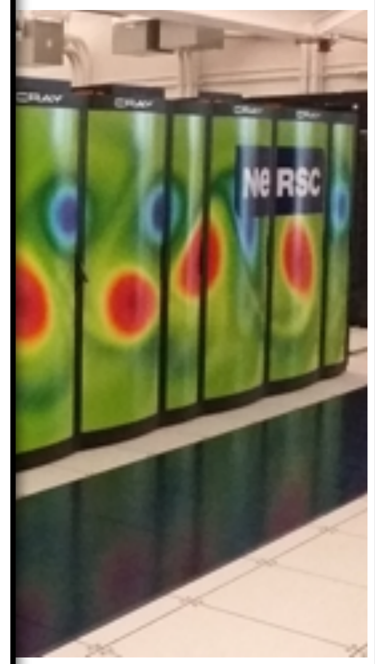
OLCF
Oak Ridge Leadership Computing Facility

- NVIDIA Kepler GPU
- 2.2GHz AMD Opteron with 32GB
- 6GB RAM on GPU
- 2.6B CPU-core-hours/year

High Performance Computers as a Bridge to HL-LHC

- 48k
- 1.6 C
- 3.1M
- 6.8B





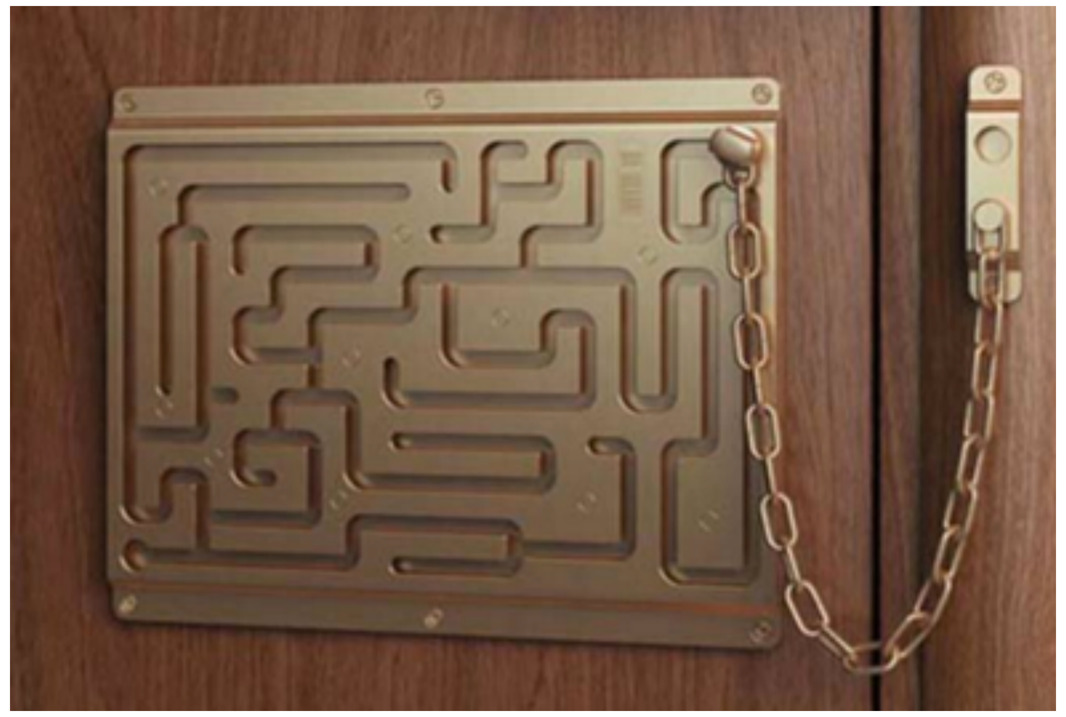
ode)

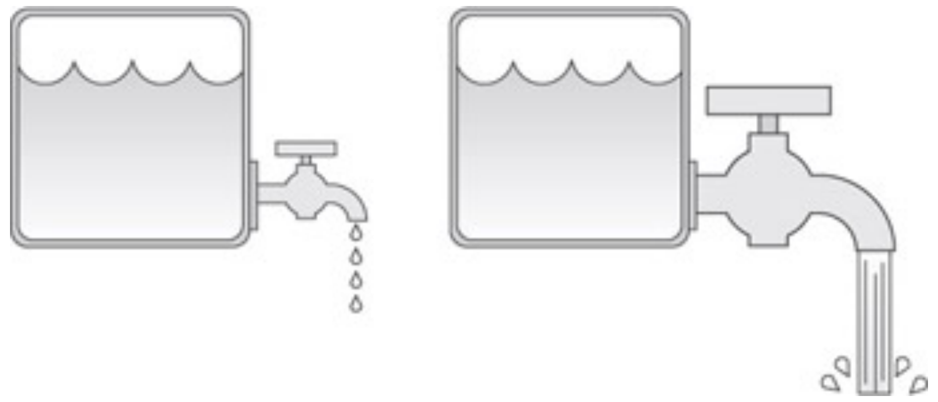
GPU

2-3 years will see a 20x increase in computing power, representing a $0.5B investment from DOE

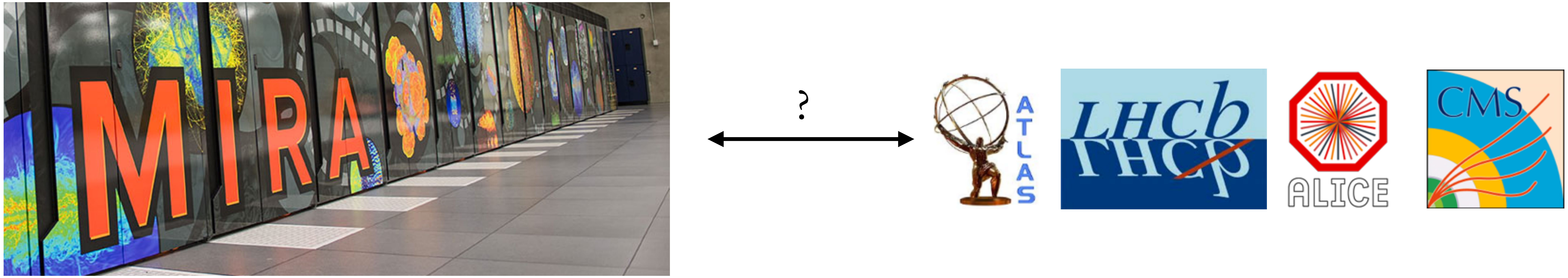
Using HPCs Requires Development

- HEP is accustom to owning and administering the machines on which our software runs.
  - We cannot become root on an HPC to install our code-stack.
- These machines typically have more security requirements.
  - A random number generator keychain is required to login to leadership class machines, like Mira and Titan.
- Large data transfers need to happen on dedicated transfer nodes, unlike the LHC Grid where transfers happen on worker nodes.
- HEP payloads typically require optimization of algorithms and workflows to run on HPCs which target more computationally intensive payloads that scale to many hundreds of computing cores.
- HPCs are not optimized for file I/O which is a bottleneck for many HEP payloads





How to use Mira for LHC Experiments?



How to use Mira for LHC Experiments?

Traditional Grid Workflow:

- Build software at CERN
- Deploy very similar architecture and linux kernel at Grid Site
- Mount remote FS on Grid Site and run binaries build at CERN

How to use Mira for LHC Experiments?

~~Traditional Grid Workflow:~~

- ~~Build software at CERN~~
- ~~Deploy very similar architecture and linux kernel at Grid Site~~
- ~~Mount remote FS on Grid Site and run binaries build at CERN~~

Mira is a PowerPC, so this doesn't work.

How to use Mira for LHC Experiments?

~~Traditional Grid Workflow:~~

- ~~Build software at CERN~~
- ~~Deploy very similar architecture and linux kernel at Grid Site~~
- ~~Mount remote FS on Grid Site and run binaries build at CERN~~

Mira Workflow:

- Build subset of software on Mira
- Run jobs there
- Upload outputs to Grid Sites for further processing

Mira is a PowerPC, so this doesn't work.

How to use Mira for LHC Experiments?

~~Traditional Grid Workflow:~~

- ~~Build software at CERN~~
- ~~Deploy very similar architecture and linux kernel at Grid Site~~
- ~~Mount remote FS on Grid Site and run binaries build at CERN~~

Mira is a PowerPC, so this doesn't work.

Mira Workflow:

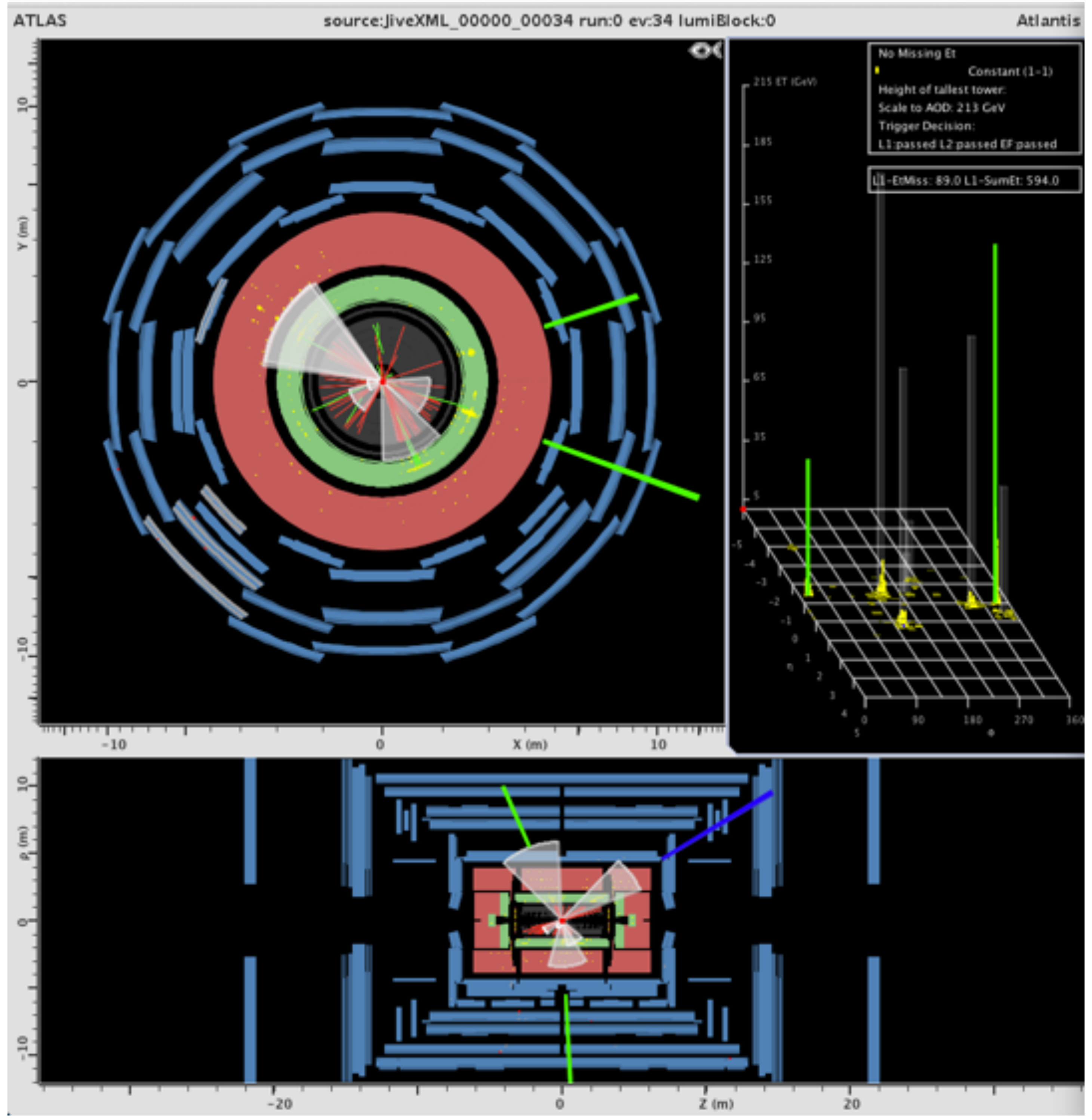
- Build subset of software on Mira
- Run jobs there
- Upload outputs to Grid Sites for further processing

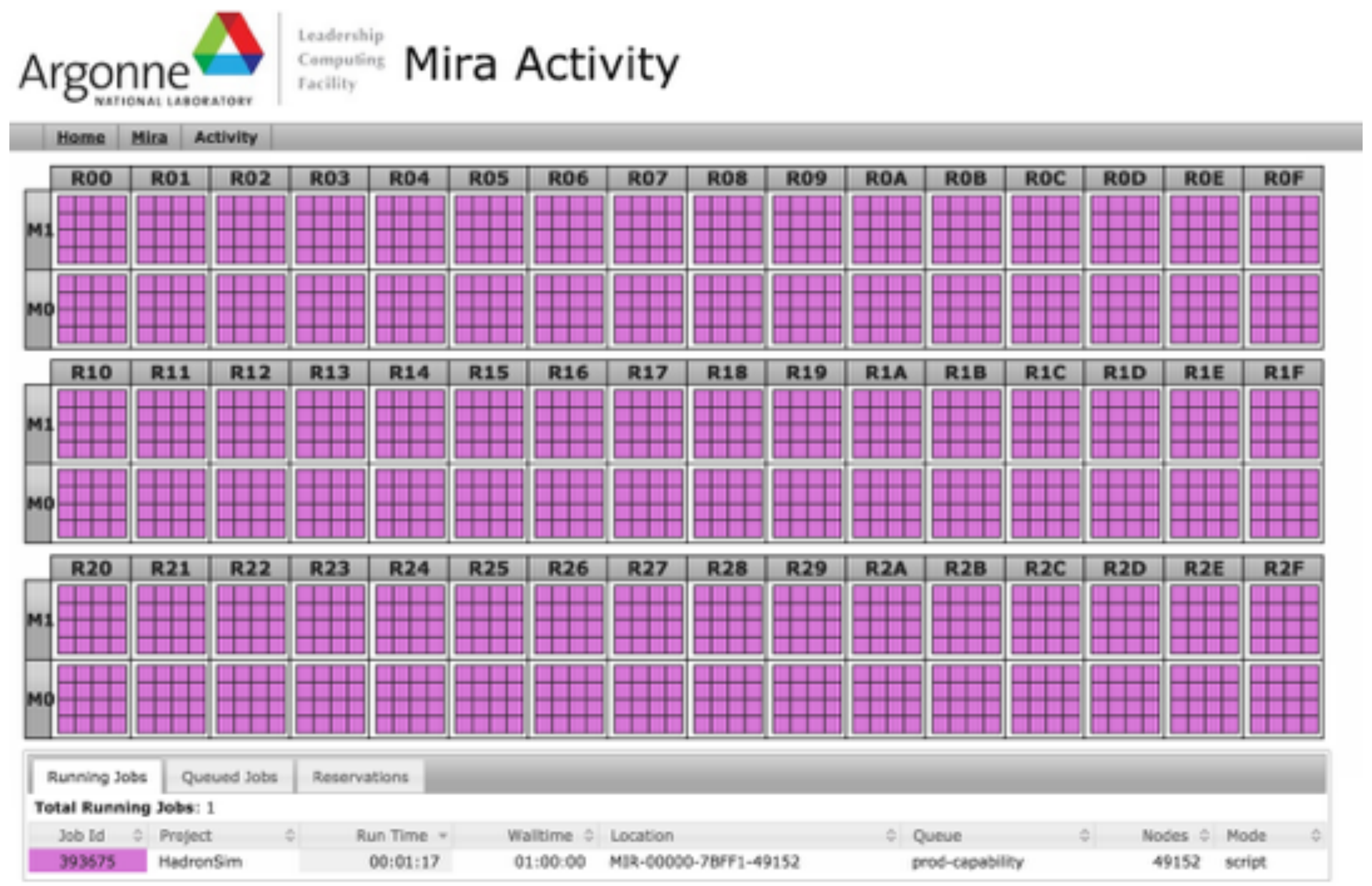
Running Event Generation:

- Experiment independent
- Easily compiled compared to experiment analysis frameworks
- Experiments already have the code to digest generator output

Job Management & Monitoring

- We needed tool for monitoring and managing all the jobs we would be running.
- The creation of 100 fb^{-1} W+5jet event requires running hundreds of jobs on Mira.
- The event generation workflow had serial and parallel steps.
  - We were not going change algorithms radically (might mess up the physics).
  - The integration process did scale on parallel machines, therefore we ran it serially on a local cluster
  - The event generation process was scaled to all 48k nodes of Mira.
- We created the HEP Edge Service (HES) to manage the many heterogeneous jobs needed to use Mira.
- HES is based on the Python Django framework taking advantage of the builtin database access and web tools.
- We used the RabbitMQ message queue system to enable remote job submission to Mira.




Argonne NATIONAL LABORATORY
Leadership Computing Facility
Home
Mira
Activity
R00 R01 R02 R03 R04 R05 R06 R07 R08 R09 R0A R0B R0C R0D R0E R0F
R10 R11 R12 R13 R14 R15 R16 R17 R18 R19 R1A R1B R1C R1D R1E R1F
R20 R21 R22 R23 R24 R25 R26 R27 R28 R29 R2A R2B R2C R2D R2E R2F
M1
M0
Running Jobs
Queued Jobs
Reservations

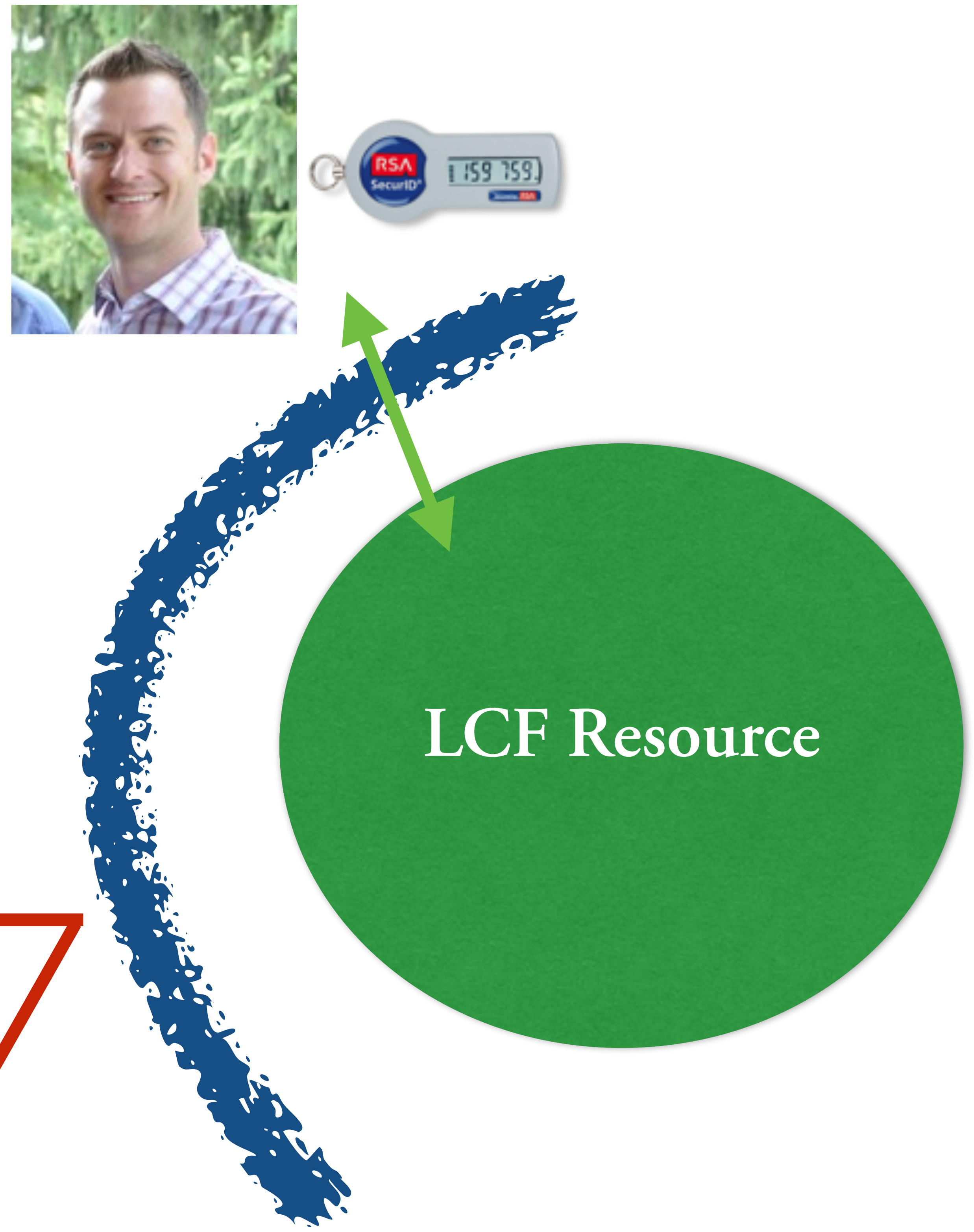
Mira Activity

Total Running Jobs: 1

Job Id	Project	Run Time	Walltime	Location	Queue	Nodes	Mode
393675	HadronSim	00:01:17	01:00:00	MIR-00000-7BFF1-49152	prod-capability	49152	script

HEP Edge Service

Typically, users need two-step authentication to access these resources so it is important we are not exposing these resources to extra risks.

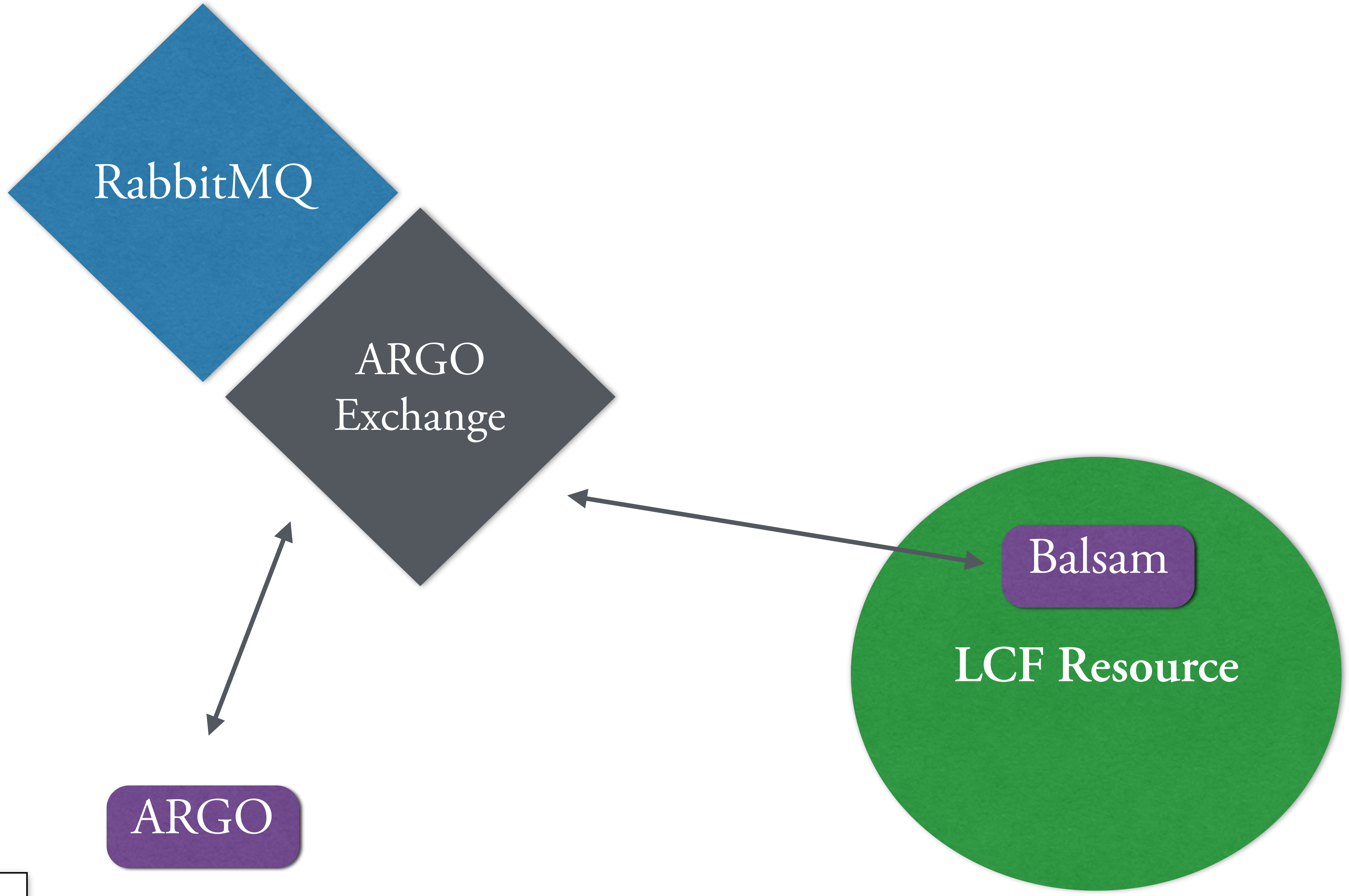


HEP Edge Service

The Balsam service runs on the resource.

Balsam
LCF Resource

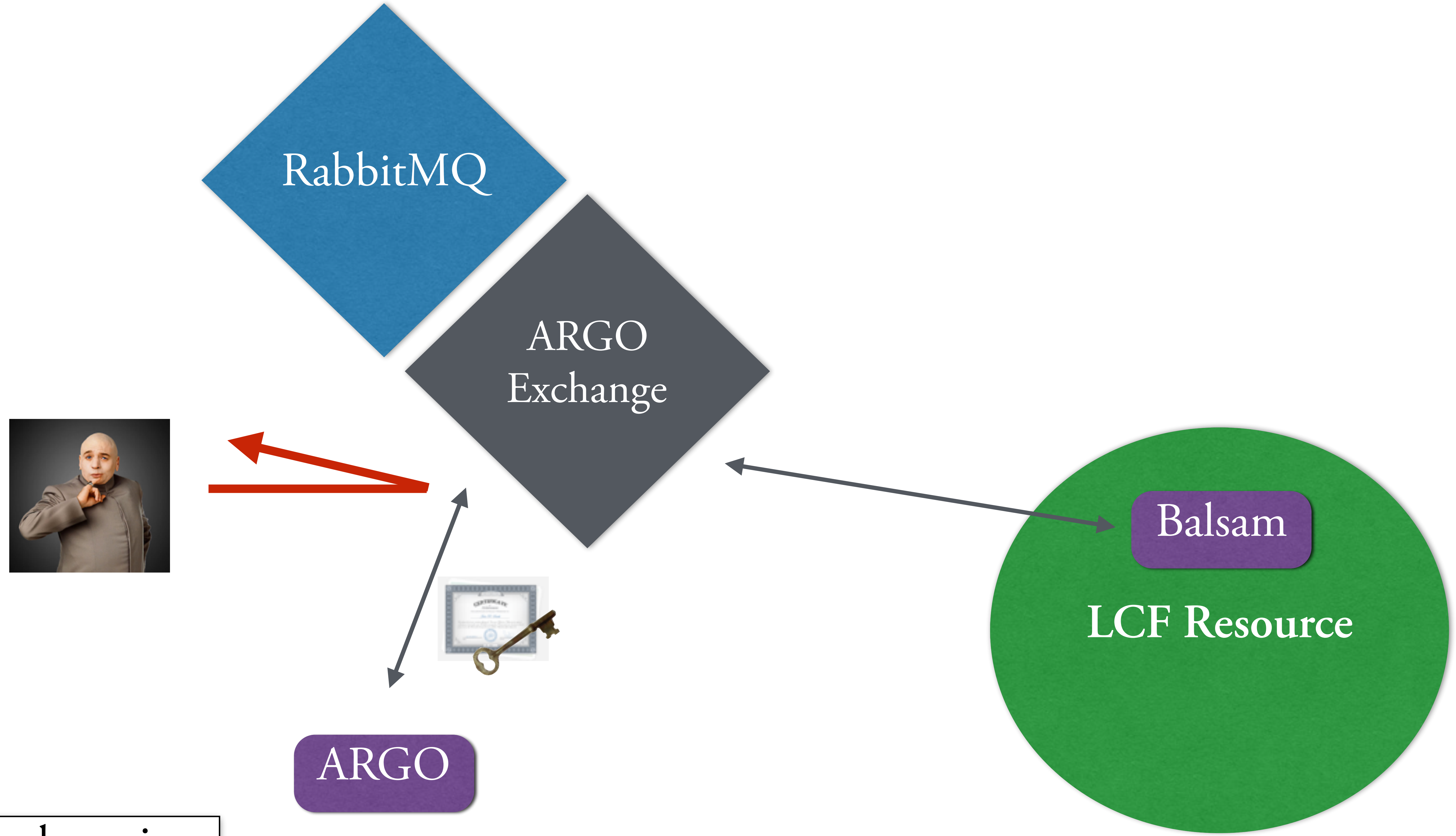
HEP Edge Service



Balsam sends and receives text messages to/from ARGO via the RabbitMQ messaging service running on an outside resource.

HEP Edge Service

- These messages are isolated inside a RabbitMQ message exchange called the "ARGO Exchange".
- A Key/Certificate pair is required to send/receive messages in this exchange.
- Only ARGO/Balsam have the key/certificate pair that are authorized to use this exchange.

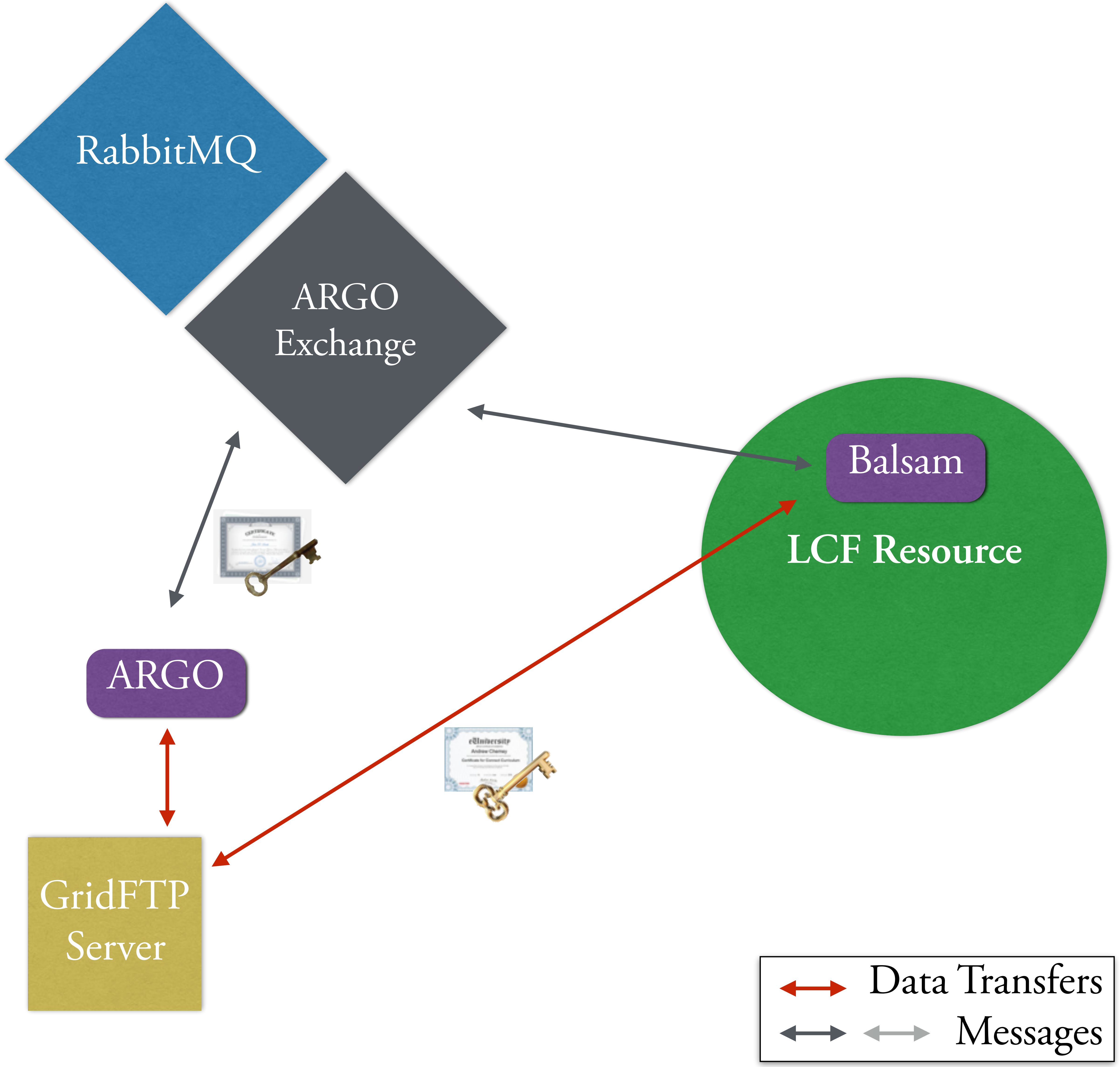


Balsam sends and receives text messages to/from ARGO via the RabbitMQ service running on an outside resource.

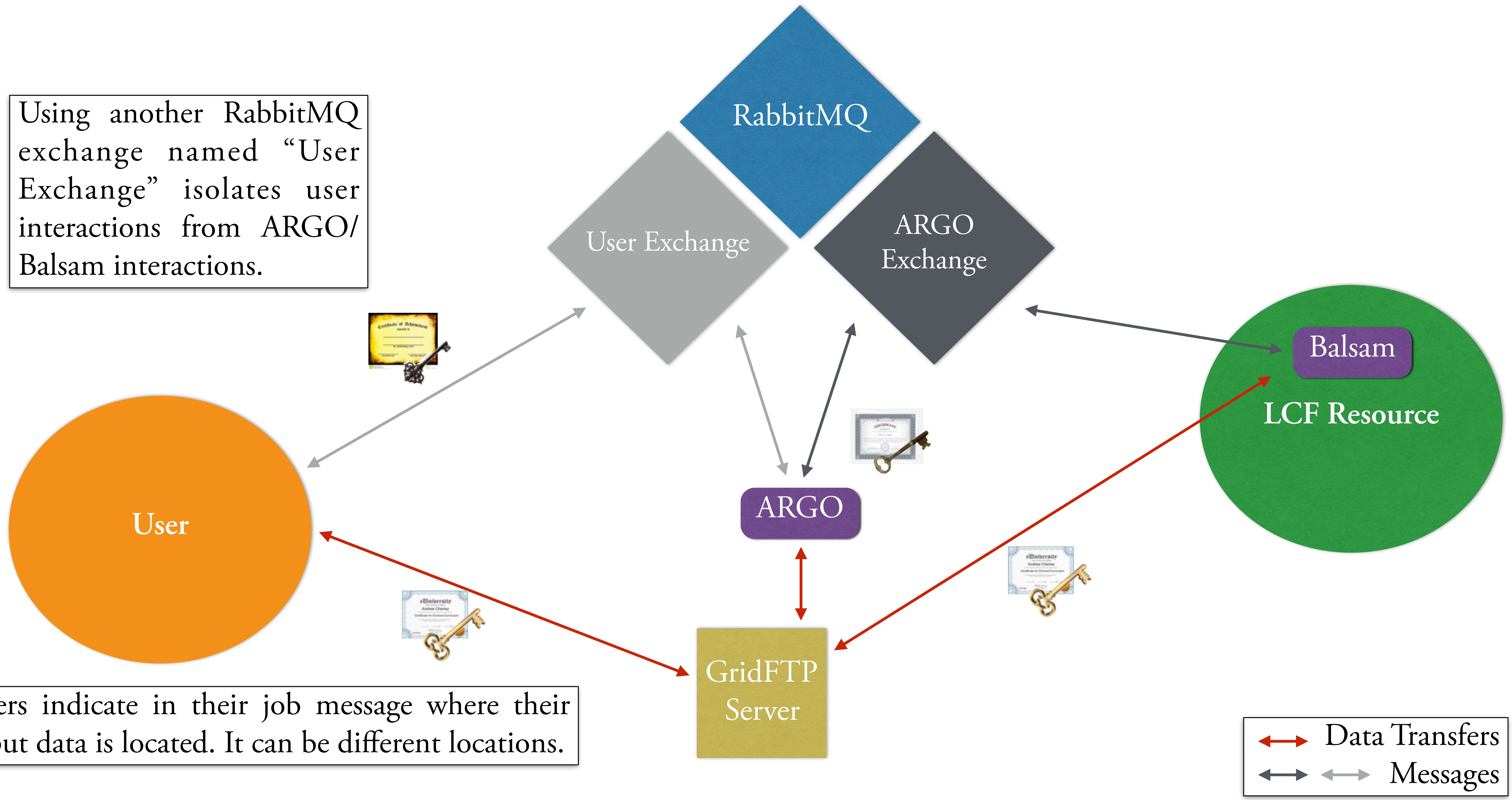
⟷ Data Transfers
⟷ ⟷ Messages

HEP Edge Service

Messages from Argo to Balsam include GridFTP URLs for handling input/output data.



HEP Edge Service



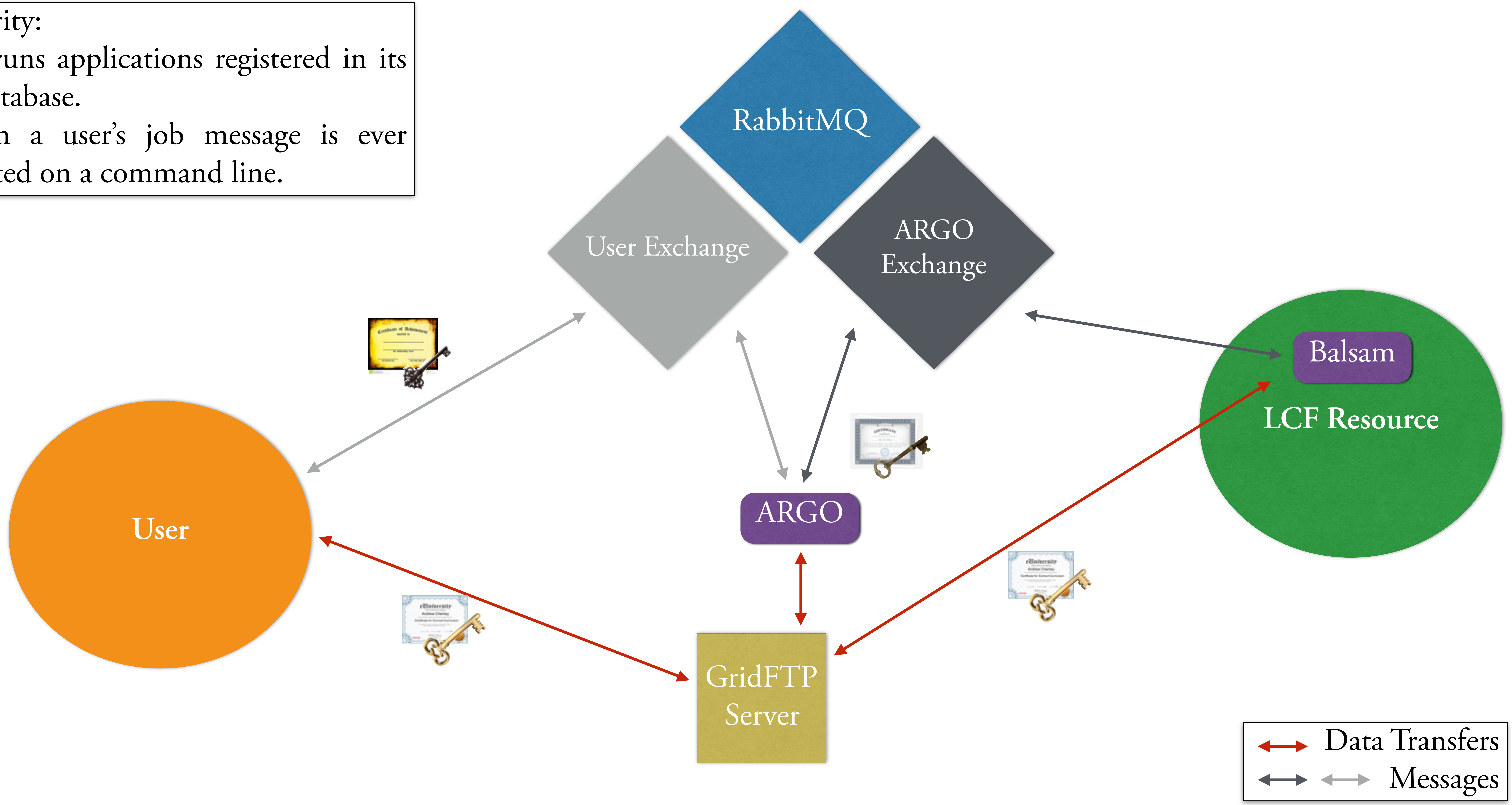
Using another RabbitMQ exchange named "User Exchange" isolates user interactions from ARGO/ Balsam interactions.

Users indicate in their job message where their input data is located. It can be different locations.

HEP Edge Service

Additional Security:

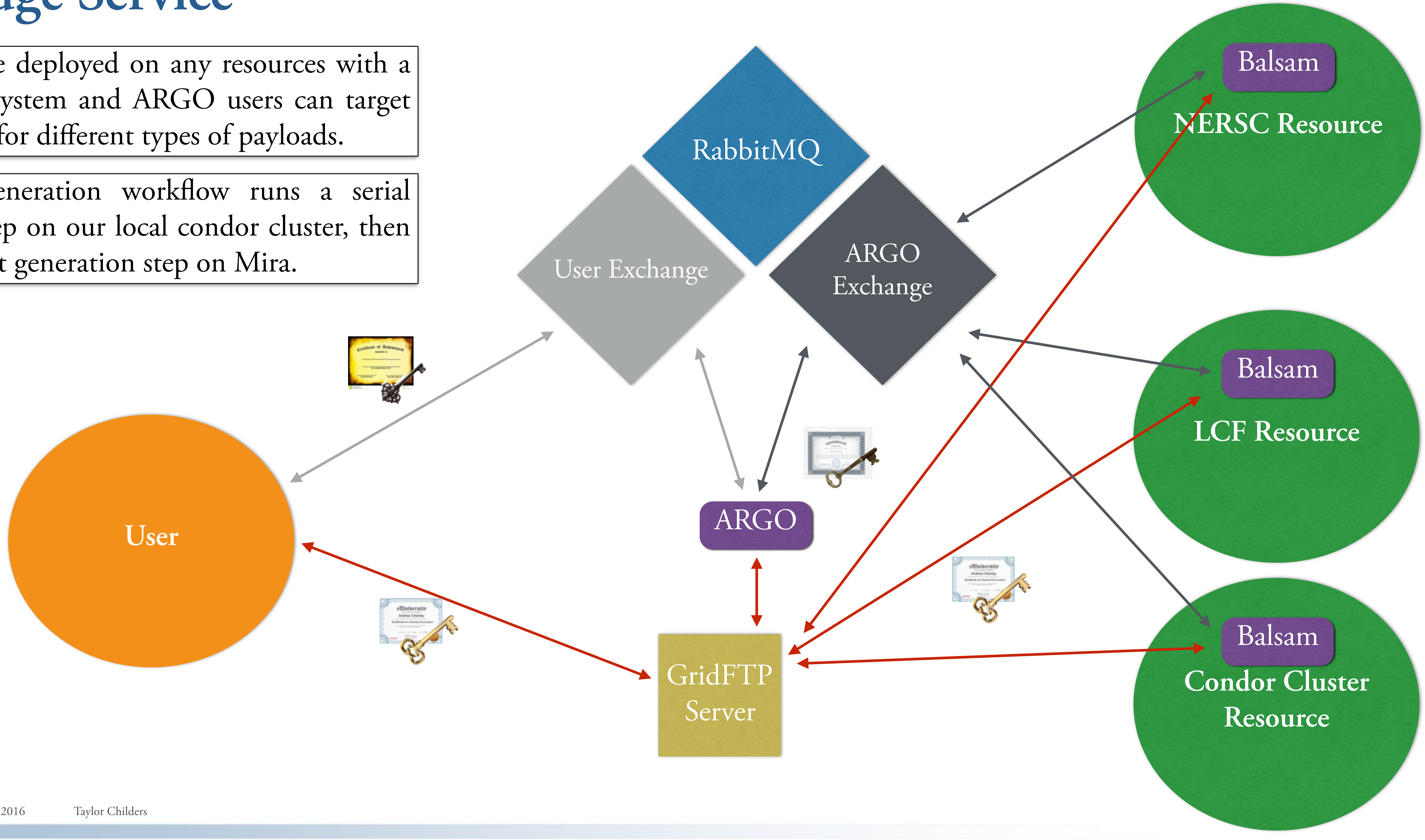
- Balsam only runs applications registered in its application database.
- Nothing from a user's job message is ever directly executed on a command line.



HEP Edge Service

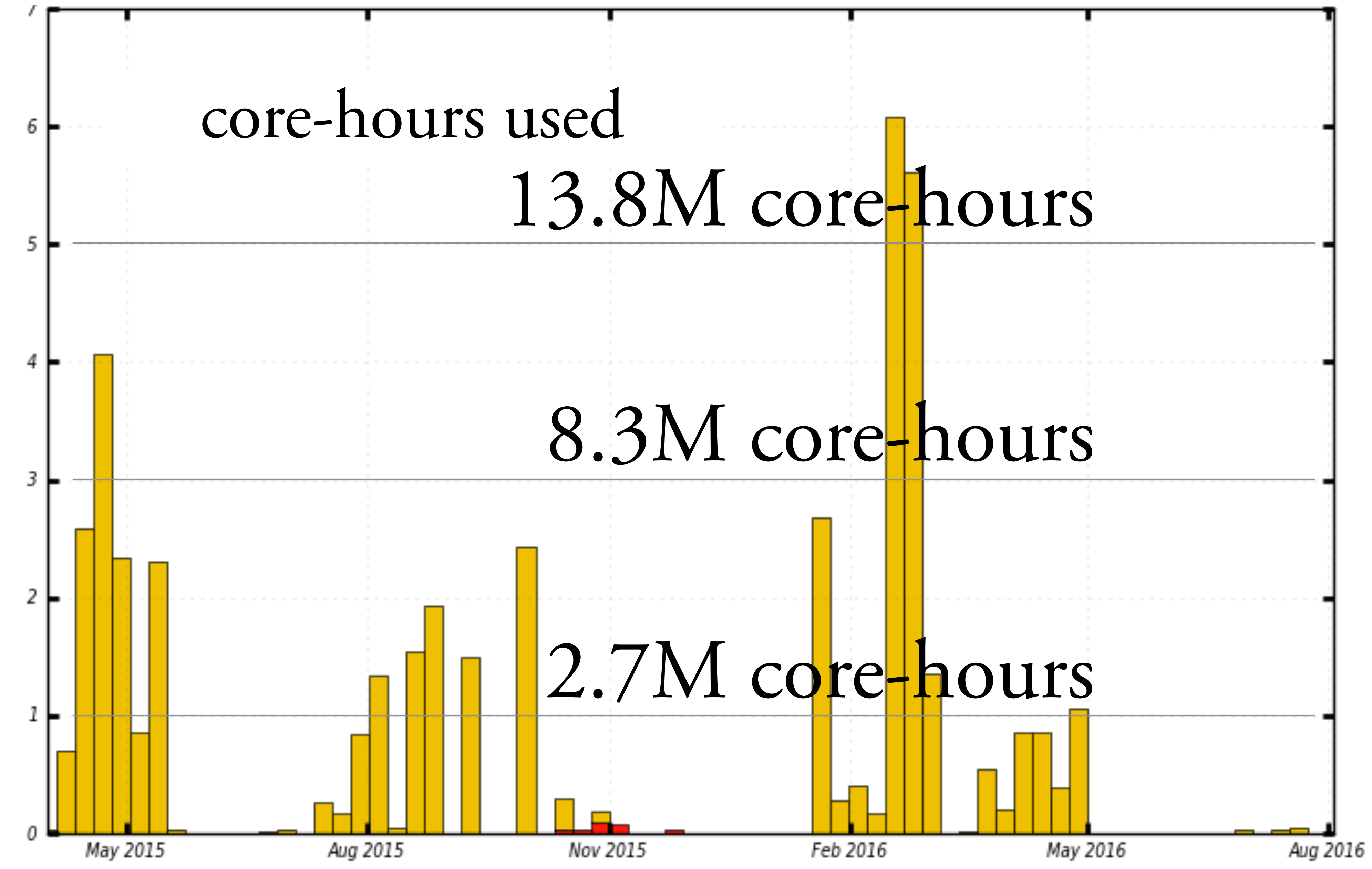
Balsam can be deployed on any resources with a batch queue system and ARGO users can target different sites for different types of payloads.

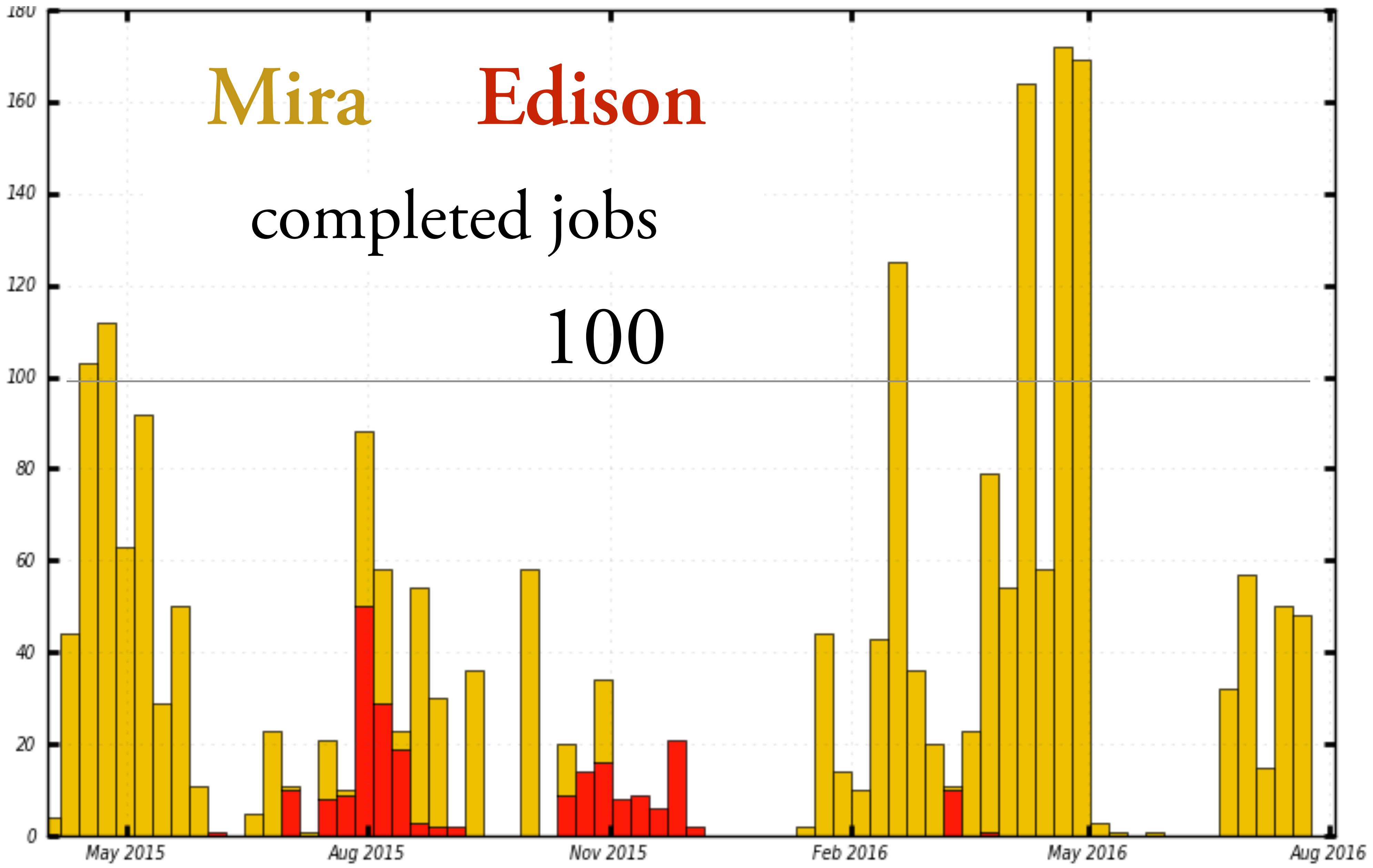
The event generation workflow runs a serial integration step on our local condor cluster, then a parallel event generation step on Mira.



HEP Edge Service Production

- Since we deployed the HEP Edge Service in April 2015, we have delivered 122M core-hours of event generation on Mira to LHC experiments.
- This included to 2000 jobs total for Mira and 230 for Edison at NERSC.
- There were not attempts to stress the system to failure and the Mira queue caused the jobs to be spread out over time.
- At the peak, the service was handling tens of jobs at once which did not stress the framework.



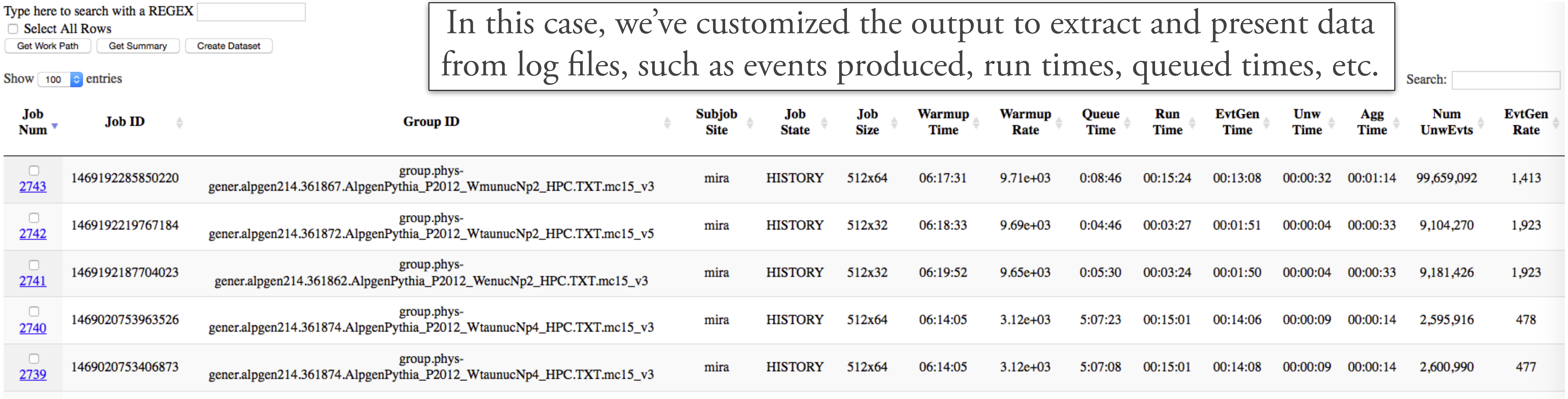


HEP Edge Service Monitoring

The Django framework is a website development package with a database interface so adding monitoring was very easy.

Listing of the jobs in the database

In this case, we've customized the output to extract and present data from log files, such as events produced, run times, queued times, etc.



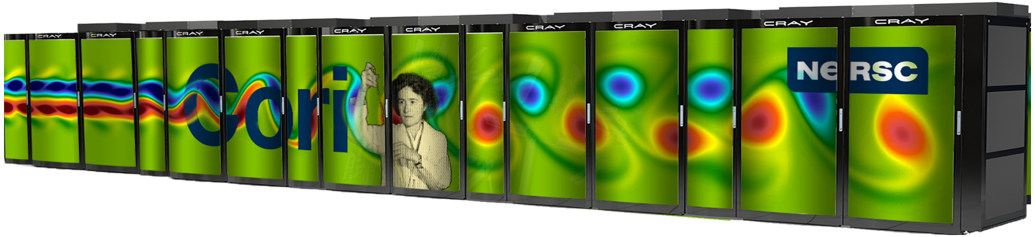
Type here to search with a REGEX

□ Select All Rows

Job Num	Job ID	Group ID	Subjob Site	Job State	Job Size	Warmup Time	Warmup Rate	Queue Time	Run Time	EvtGen Time	Unw Time	Agg Time	Num UnwEvts	EvtGen Rate
□ 2743	1469192285850220	group.phys-gener.alpgen214.361867.AlpgenPythia_P2012_WmunucNp2_HPC.TXT.mc15_v3	mira	HISTORY	512x64	06:17:31	9.71e+03	0:08:46	00:15:24	00:13:08	00:00:32	00:01:14	99,659,092	1,413
□ 2742	1469192219767184	group.phys-gener.alpgen214.361872.AlpgenPythia_P2012_WtaunucNp2_HPC.TXT.mc15_v5	mira	HISTORY	512x32	06:18:33	9.69e+03	0:04:46	00:03:27	00:01:51	00:00:04	00:00:33	9,104,270	1,923
□ 2741	1469192187704023	group.phys-gener.alpgen214.361862.AlpgenPythia_P2012_WenucNp2_HPC.TXT.mc15_v3	mira	HISTORY	512x32	06:19:52	9.65e+03	0:05:30	00:03:24	00:01:50	00:00:04	00:00:33	9,181,426	1,923
□ 2740	1469020753963526	group.phys-gener.alpgen214.361874.AlpgenPythia_P2012_WtaunucNp4_HPC.TXT.mc15_v3	mira	HISTORY	512x64	06:14:05	3.12e+03	5:07:23	00:15:01	00:14:06	00:00:09	00:00:14	2,595,916	478
□ 2739	1469020753406873	group.phys-gener.alpgen214.361874.AlpgenPythia_P2012_WtaunucNp4_HPC.TXT.mc15_v3	mira	HISTORY	512x64	06:14:05	3.12e+03	5:07:08	00:15:01	00:14:08	00:00:09	00:00:14	2,600,990	477

Data Intensive Computing at NERSC

- One of NERSC's three operational pillars, group of ~10 people dedicated to optimizing data intensive workloads at NERSC
- Main vehicle is Cori
  - Dedicated data partition
  - Phase II will be KNL



- Cori's data friendly aspects
  - **Shifter**: Docker-like functionality
  - **Burst Buffer**: Super fast I/O layer
  - 1632 Haswell nodes with 128 GB of RAM
  - Lustre file system with ~30 PB of space and 700 GB/s bandwidth
  - Computes can access outside world
  - Shared queue to support single core jobs

Shifter: Docker for HPC

- **Docker: open source, automated container deployment service**
  - Docker containers wrap up a piece of software in a complete file system that contains everything it needs to run (code, environment, system tools, and libraries)
  - Guaranteed to operate the same, regardless of the environment in which it is running
- **NERSC has implemented Docker-like container technology through a new software package called Shifter**
  - Supports Docker and other images (vmware, ext4, squashfs, etc.)
  - Users can upload custom images in desired OS
  - Tied into the batch system
  - Intended for complex code stacks and codes with shared libraries

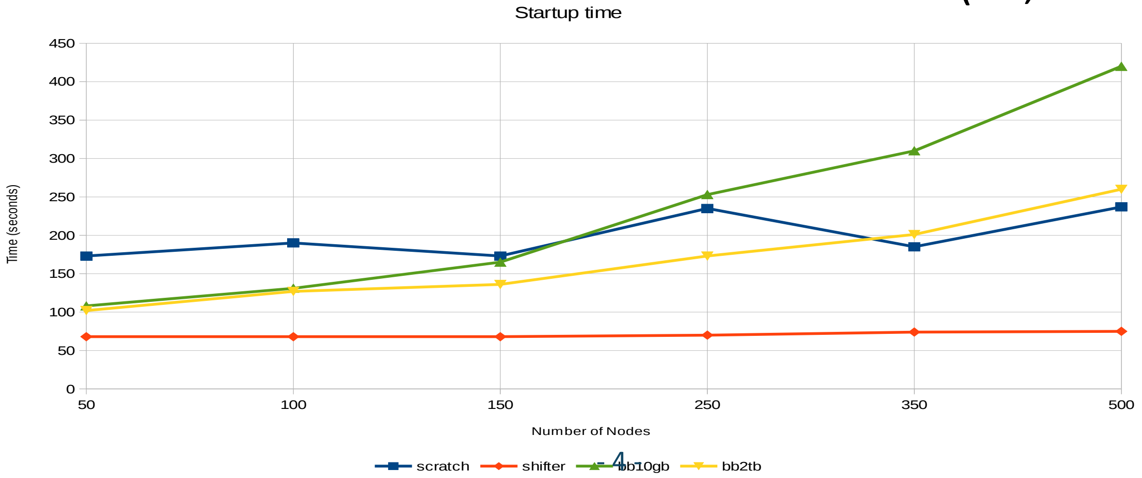
Work of Doug Jacobsen at NERSC

Shifter Example: CVMFS on Cray

- **Problem: No way to run CVMFS in standard HPC (Cray) environment**
  - Compute nodes have a highly optimized, stripped-down Linux environment
  - No local disk
  - Root access only in very special circumstances
- **Can't directly containerize it**
  - Docker images that run CVMFS require elevated kernel packages, not possible on compute nodes
- **Solution: Create a shifter image with CVMFS unpacked onto it**
  - Monster sized images: 300 – 500 GB after compression and deduping

Running CVMFS Shifter Image

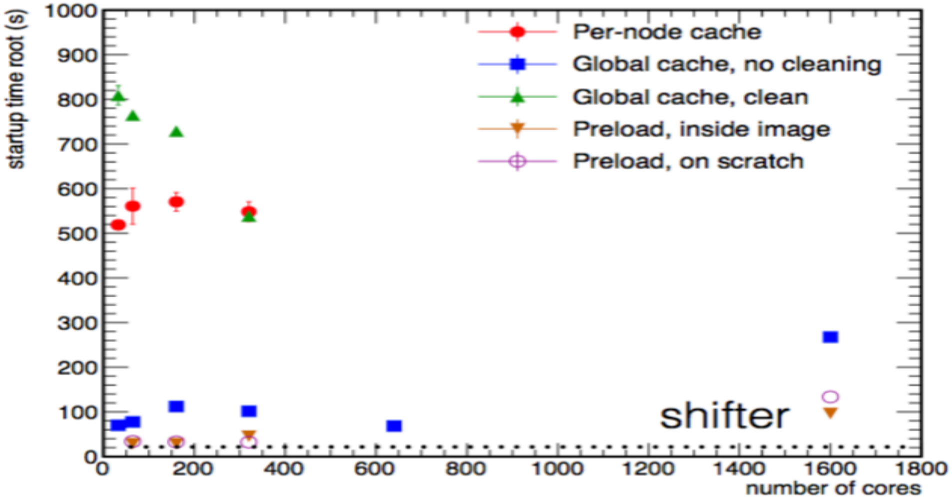
- **Use Shifter to load job**
  - Add a single flag to batch script “--image=<image name>”
  - ATLAS cvmfs repository is found at /cvmfs/atlas.cern.ch like normal
- **Tested with ATLAS G4 simulations and Analysis Software (QuickAna)**
  - Simulation load times scale well out to 500 nodes (16,000 cores)



Work of Vakho Tsulaia at LBNL

CVMFS and parrot in Shifter

- **Big images can only be made ~once / day**
- **Investigating using parrot to load from alien cache**
  - See comparable load times with CVMFS cache on parallel file system



Work of Markus Fasel at LBNL

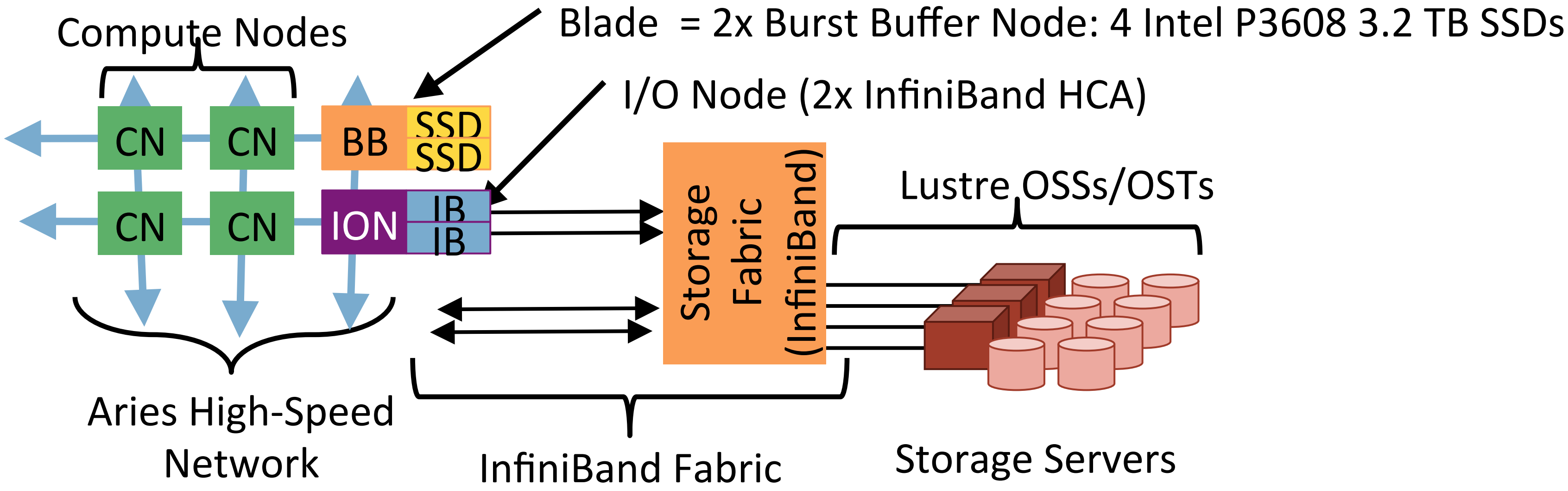
Future Shifter Developments

- **A number of HEP and astronomy groups are already using Shifter at NERSC**
  - CMS, ATLAS, LSST, LCLS
  - Cray is working to make Shifter a mainstream capability for their systems
- **Shifter has been approved to be released as open source through a BSD license**
  - The intent is that others can download it and use it at their centers
  - **Shifter-hpc google group** for those interested in installing the framework on their systems

Burst Buffer: Accelerating I/O at NERSC

- **Burst Buffer (Phase 1) 920TB on 144 BB nodes**
- **NVRAM-based storage**
  - For bandwidth spinning disk is more expensive than SSD
  - Handle I/O bandwidth spikes without increasing size of PFS
  - Underlying media supports challenging I/O patterns
  - File systems on demand scale better than large POSIX PFS
  - Staging to PFS asynchronously
- **Experimental HEP applications have challenging I/O patterns**
  - High IOPS – better match for SSD than spinning disk

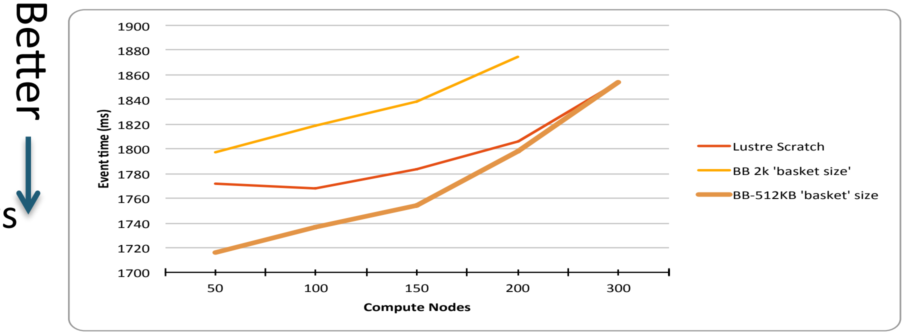
Burst Buffer Architecture

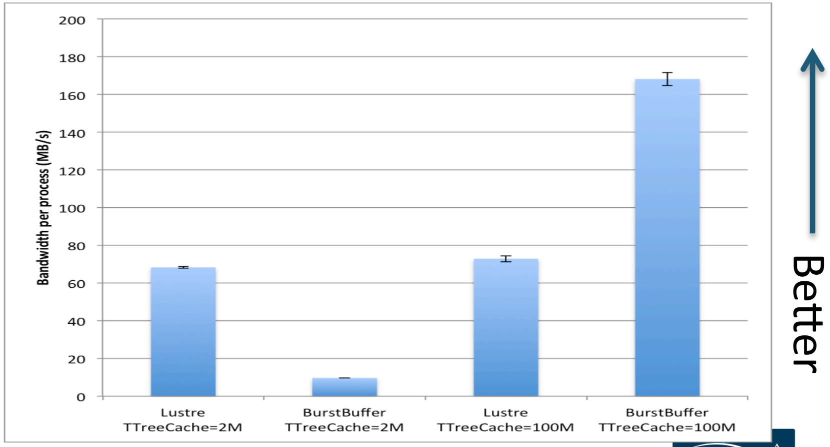

Compute Nodes
Blade = 2x Burst Buffer Node: 4 Intel P3608 3.2 TB SSDs
I/O Node (2x InfiniBand HCA)
CN
CN
BB
SSD
SSD
CN
CN
ION
IB
IB
Storage Fabric (InfiniBand)
Lustre OSSs/OSTs
Aries High-Speed Network
InfiniBand Fabric
Storage Servers

- **DataWarp software manages storage**
  - Integrated with SLURM batch system
  - Allocates portions of storage to users on a per-job basis
- **Users see a POSIX file system**
- **File system can be striped across multiple nodes**

ATLAS Software on the Burst Buffer

- **ATLAS simulations with Yoda**
  - Scales well out to 9600 cores with minor tweaks
    - Write block size increased and small logs files moved
- **ATLAS analysis with QuickAna**
  - Analyzed a 475 GB xAOD dataset
  - Burst Buffer bandwidth a factor of 2 higher than Lustre
  - Performance expected to improve once client side caching is enabled
  - Future plans: Scale out to O(10k) cores and O(10T) data





Conclusion

- **New frameworks and innovations are making running HEP workloads easier at NERSC and ALCF**
  - HEP Edge Service: Gateway to HPC, served more than 100 M core hours
  - Shifter: Portable environment
  - Burst Buffer: Super fast I/O
  - Many other cool projects:

    Software Defined Networking, **Machine Learning**

Saturday Poster Session: “Exploring Raw HEP Data using Deep Neural Networks”, E. Racah et al.

- **HPC will be a critical tool for LHC-HL and other future experiments**

Collaborators

NERSC / LBNL : Wahid Bhimji, Deborah Bard, Prabhat, Jeff Porter, Markus Fasel, Vakhtang Tsulaia, Steve Farrell

ALCF: Doug Benjamin (DukeU), Tom LeCompte (Argonne), Tom Uram (Argonne)

How Shifter Works

User loads image
“docker pull <image_name>”

Batch system
prolog stages
image on nodes

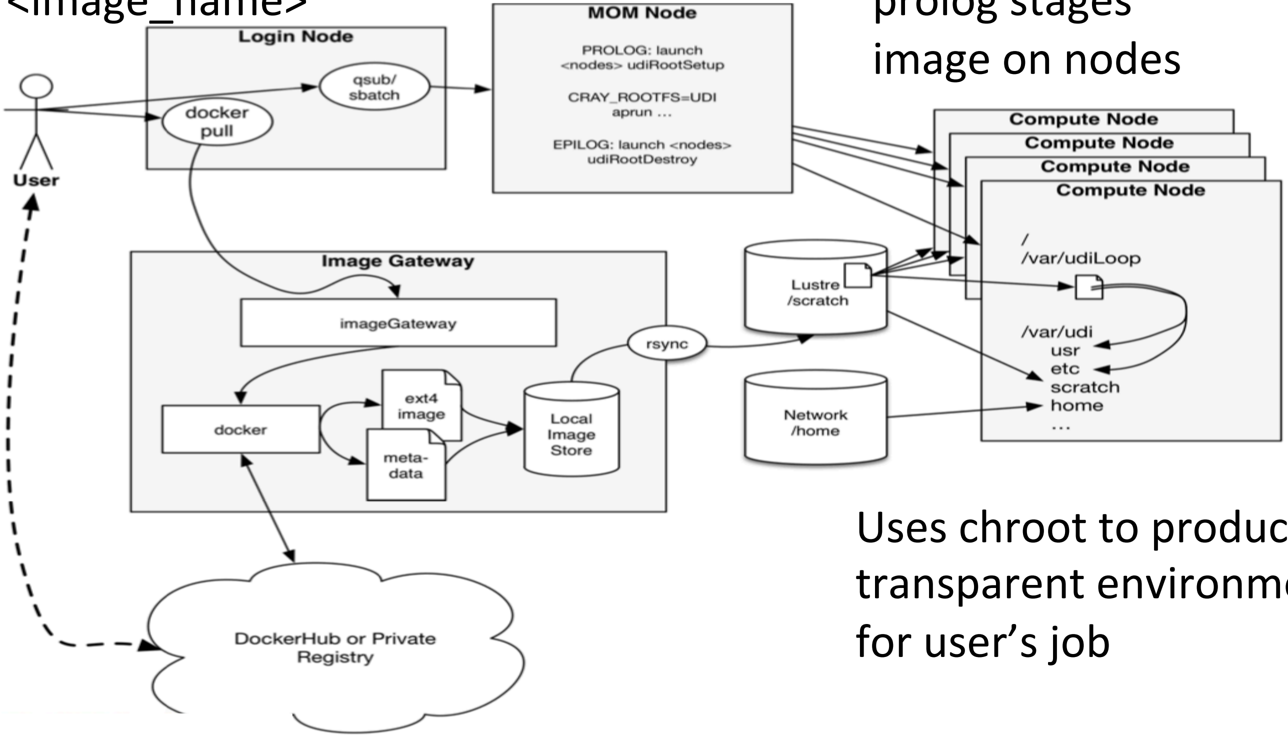


Image is stored
by gateway
service

Uses chroot to produce
transparent environment
for user’s job

Burst Buffer Performance

- **Burst Buffer is doing well against benchmark performance targets**
  - Work on-going to improve MPIO shared file write
  - Out-performs Lustre

	IOR Posix FPP		IOR MPIO Shared File		IOPS	
	Read	Write	Read	Write	Read	Write
Best Measured (140 Burst Buffer Nodes : 1120 Compute Nodes; 4 ranks/node)*	905 GB/s	873 GB/s	803 GB/s	351GB/s	12.6 M	12.5 M
Lustre (peak – 24 OSTs: 930 compute nodes, 4 ranks/node; 4 MB transfer)	708 GB/s	751 GB/s	573 GB/s	223 GB/s	-	-

**Bandwidth tests: 8 GB block-size 1MB transfers IOPS tests: 1M blocks 4k transfer*