Job Submission and Debugging Your Batch Job

Vito Di Benedetto

Intensity Frontier Computing Summer School

June 21, 2021

Outline

- Why use computing grid
- Submit grid jobs
- JobSub overview
- Monitoring and debugging jobs

Why use computing grid

- Nowadays users need to process tons of data
  - It is not practical for users to accomplish those tasks using their own workstation
- Most of the tasks used to process data can be organized in smaller *units* that can be executed within hours
- It is convenient to have access to a computing farm/grid with thousands of **worker nodes** available to users
- Each of those *unit*/**job** can be processed by a **worker node**
- We need a way to dispatch jobs to worker nodes
  - **HTCondor** is our friend
- SCD developed **JobSub**, a tool that works as an interface to HTCondor
  - JobSub makes it easier to submit jobs to the computing grid

How to submit grid jobs

- I would say: *use JobSub*
- but each experiment developed their submission tools to make it easier to automatize job submission
- However all experiment submission tools are based on JobSub
- I'm not going to provide details for specific experiment submission tools
- I'll provide an overview of JobSub commands
  - Most of JobSub submission features are implemented by experiments in their submission tools
- Resources:
  - https://cdcvs.fnal.gov/redmine/projects/jobsub/wiki/Using_the_Client
  - https://cdcvs.fnal.gov/redmine/projects/jobsub/wiki/Frequently_Asked_Questions

JobSub overview

- JobSub is made available through the UPS package *jobsub_client*
- In most cases the setup of the experiment code pulls in JobSub as dependency, if not JobSub can be setup this way:

```
source /cvmfs/fermilab.opensciencegrid.org/products/common/etc/setups.sh
setup jobsub_client
```

- Check you have valid kerberos ticket to get JobSub authenticated

```
klist -s; echo $?
0
```

- The most common used JobSub commands are:

```
- jobsub_submit
- jobsub_q
- jobsub_fetchlog
- jobsub_rm
- jobsub_hold
```

jobsub_submit --> submitting jobs

- Submitting a simple test job:

```
jobsub_submit -G <group> --role=Analysis -N 1 --expected-lifetime=1h --memory=100MB \
--disk=3GB --cpu=1 --resource-provides=usage_model=DEDICATED,OPPORTUNISTIC,OFFSITE \
file:///nashome/v/vito/jobsub_test/probe burn=10
```

- Accounting options:
  -G group, this corresponds to your experiment
  --role user VOMS role, default role is Analysis
- **-N** number of jobs to submit in the cluster
- Resources options:
  --expected-lifetime, --memory, --disk, --cpu request this much for max run time, memory disk and number of CPUs, the format is NUMBER[units].
  --resource-provides=usage_model this controls where jobs are allowed to run. **DEDICATED** means use your experiment FermiGrid quota, **OPPORTUNISTIC** means use idle FermiGrid resources beyond your experiment quota if they are available, and **OFFSITE** means use non-Fermilab resources. You can combine them in a comma-separated list.
  Default value is **DEDICATED,OPPORTUNISTIC,OFFSITE**, this ensures maximum resource availability and will get your jobs started the fastest.
- **file:///path/to/user/script [script arguments]**
- **jobsub_submit -G <group> --help** provides full help.
- the **probe** script of this example runs a CPU intensive process for 10 s.
  run **/nashome/v/vito/jobsub_test/probe -h** to get details for the probe script options.
- **jobsub_submit output:**

```
/fife/local/scratch/uploads/fermilab/vito/2021-06-18_153211.537104_9806
/fife/local/scratch/uploads/fermilab/vito/2021-06-18_153211.537104_9806/probe_20210620_153212_2694610_0_1_.cmd
submitting....
Submitting job(s).
1 job(s) submitted to cluster 45385219.
JobsubJobId of first job: 45385219.0@jobsub01.fnal.gov
Use job id 45385219.0@jobsub01.fnal.gov to retrieve output
```

jobsub_q --> check your jobs status in the queue

- Check your jobs status in the queue:

```
jobsub_q –G <group> --user ${USER} --jobid <JobID>
```

- jobsub_q output:

```
JOBSUBJOBID                 OWNER SUBMITTED    RUN_TIME   ST PRI SIZE  CMD
45385219.0@jobsub01.fnal.gov vito  06/18 15:32  0+00:02:32 R  0   0.0   probe_20210618_153212_2694610_0_1_wrap.sh
1 jobs; 0 completed, 0 removed, 0 idle, 1 running, 0 held, 0 suspended
```

jobsub_submit --> submitting jobs (2)

- Submitting few more test jobs:

```
jobsub_submit -G <group> --role=Analysis -N 1 --expected-lifetime=1h --memory=100MB \
--disk=3GB --cpu=1 --resource-provides=usage_model=DEDICATED,OPPORTUNISTIC,OFFSITE \
file:///nashome/v/vito/jobsub_test/probe burn=10,error=1
```

  - This job will fail with exit code 1

```
jobsub_submit -G <group> --role=Analysis -N 1 --expected-lifetime=1h --memory=100MB \
--disk=3GB --cpu=1 --resource-provides=usage_model=DEDICATED,OPPORTUNISTIC,OFFSITE \
file:///nashome/v/vito/jobsub_test/probe burn=10,memory=500
```

  - This jobs will use 500MB of memory, but we requested only 100MB of memory. The job will be held for exceeding requested usage.
    This can be avoided using the **autorelease feature**

```
jobsub_submit -G <group> --role=Analysis -N 1 --expected-lifetime=1h \
-l '+FERMIHTC_AutoRelease=True' -l '+FERMIHTC_GraceMemory=1024' --memory=100MB \
--disk=3GB --cpu=1 --resource-provides=usage_model=DEDICATED,OPPORTUNISTIC,OFFSITE \
file:///nashome/v/vito/jobsub_test/probe burn=10,memory=500
```

  - If the a job in this cluster exceeds requested memory, it is hold, after few minutes the job is automatically released with an updated requested memory that is increased by the **FERMIHTC_GraceMemory** amount.
  - There is a similar option to increase job run time: **FERMIHTC_GraceLifetime**

- In many cases job memory usage and run time can have a long queue. Always make sure to require memory and run time that cover up to 95% of your jobs, then use autorelease feature to handle remaining jobs. This allows a more efficient use of resources.

jobsub_submit --> submitting jobs (3)

- Submitting one more test jobs:

```
jobsub_submit -G <group> --role=Analysis -N 1 --expected-lifetime=1h --memory=100MB \
--disk=3GB --cpu=1 --resource-provides=usage_model=DEDICATED,OPPORTUNISTIC,OFFSITE \
--tar_file_name dropbox:///nashome/v/vito/jobsub_test/probe.tar.gz \
-l '+SingularityImage=\"/cvmfs/singularity.opensciencegrid.org/fermilab/fnal-wn-sl7:latest\"' \
--append_condor_requirements='(TARGET.HAS_Singularity==true)' \
file:///nashome/v/vito/jobsub_test/probe burn=10
```

- Here we have used these other options:

  --tar_file_name dropbox:///path/to/tarball

  This option is useful to use custom code in the job. By default the code is deployed to the job through a dedicated CVMFS repository using **Rapid Code Distribution Service (RCDS)**

  -l '+SingularityImage=…' --append_condor_requirements='(TARGET.HAS_Singularity==true)'

  These option requires the job to run inside a Singularity container.
  This allows jobs to run in the same environment on all worker nodes for all sites.

jobsub_fetchlog --> check your jobs log

- Check your jobs log:
  `jobsub_fetchlog -G <group> --jobid <JobId> --destdir <Log Dir for JobID>`
- For example:
  `jobsub_fetchlog -G fermilab --jobid 123456.0@jobsub0N.fnal.gov --destdir 123456`
- Download all logs for job cluster 123456.0@jobsub0N.fnal.gov as tarball and unwind it in your chosen destdir.

jobsub_rm --> remove selected jobs

- Remove selected jobs:

```
jobsub_rm –G <group> --user ${USER} --jobid <JobId>
```

- this command removes selected jobs from the queue, be cautious and double check jobs to remove.

jobsub_hold --> hold selected jobs

- Hold selected jobs:

```
jobsub_hold –G <group> --user ${USER} --jobid <JobId>
```

- this command holds selected jobs from the queue, be cautious and double check jobs to hold.

Best practice in grid jobs and more

- A good list of best practice for grid jobs and more is available here: https://cdcvs.fnal.gov/redmine/projects/fife/wiki/Best_Practice
- When creating a new workflow or making changes to an existing one, **ALWAYS test** with a single job first. Then go up to 10, etc. Don't submit thousands of jobs immediately and expect things to work.
- **ALWAYS** be sure to prestage your input datasets before launching large sets of jobs.
- Use **RCDS** to run your custom code on the grid
- Be careful about placing your output files. NEVER place more than a few thousand files into any one directory inside dCache. That goes for all type of dCache (scratch, persistent, resilient, etc).
- Use xrootd when opening files interactively; this is much more stable than simply doing root /pnfs/dune/…
- **NEVER** do hadd on files in /pnfs areas unless you're using xrootd. This can cause severe performance degradation.

Monitoring jobs – email report

The following cluster of your jobs recently completed on FermiGrid/Fifebatch. The information below is provided to help you better understand the resource requirements of your batch jobs, improve their performance, and minimize the time to completion (a.k.a. get you results faster).

Cluster	43554347@jobsub01.fnal.gov
Number of Jobs	482
Submitted	2021-05-04 23:25:15 +0000 UTC
Owner/Group	vito / uboone (host/jenkins02.fnal.gov@FNAL.GOV)
Command	workernode_wrapper_script_Data_Data_Cosmic_reco2_ana_data_lar_ci_sl7_923.sh
Requested Memory	5000 MiB
Requested Disk	50.0 GiB
Expected Wall Time	10h6m0s

View this cluster on Fifemon

Average time waiting in queue: 22m27s

Success rate (% jobs with exit code 0): 91.5%

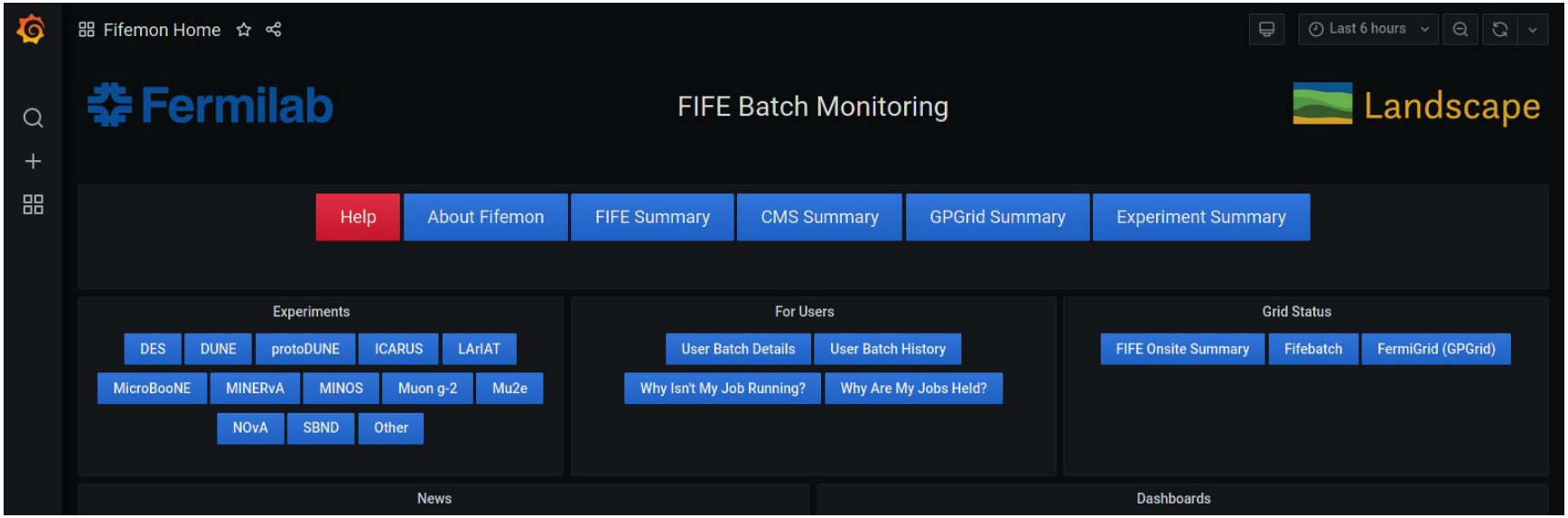
Used	Min	Max	Avg
Memory	807.6 MiB	4895.5 MiB	2821.6 MiB
Disk	0.0 GiB	5.5 GiB	0.2 GiB
Wall Time	0s	7h53m7s	1h59m36s
CPU Time	2m5s	7h48m28s	1h40m58s

Efficiency	Min	Max	Avg
Memory	16.5%	100.3%	57.8%
Disk	0.0%	15.8%	0.5%
CPU	9.5%	100.0%	84.4%
Time	0.0%	93.5%	22.3%

Exit Code	# Jobs
0	441
1	7
11	33
6	1

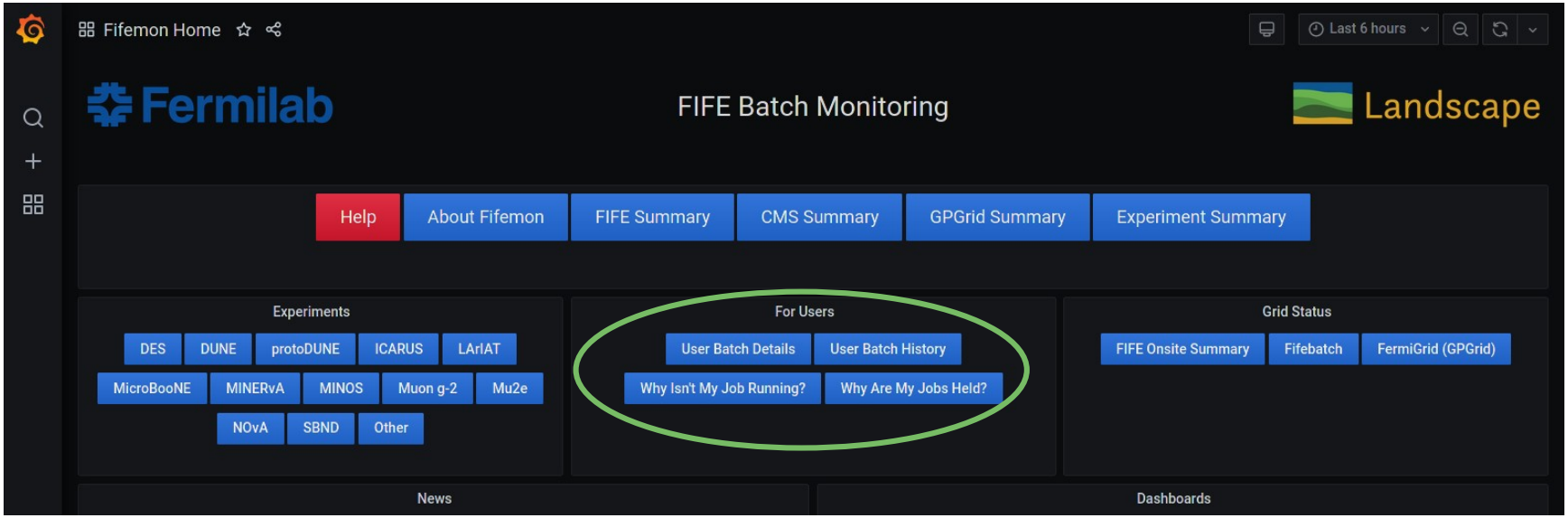
Monitoring jobs – Landscape

- Landscape



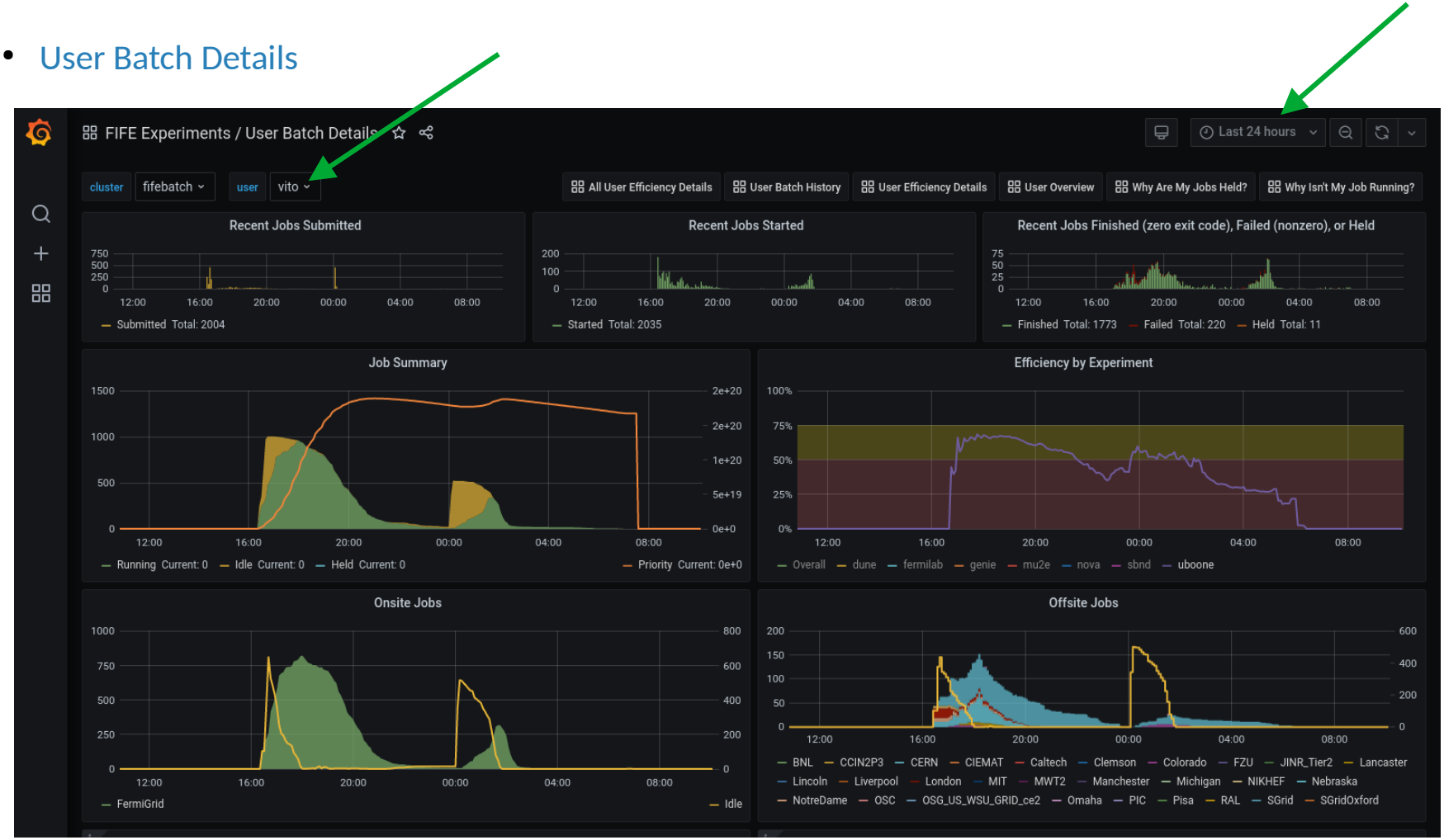
Monitoring jobs – Landscape

- Landscape



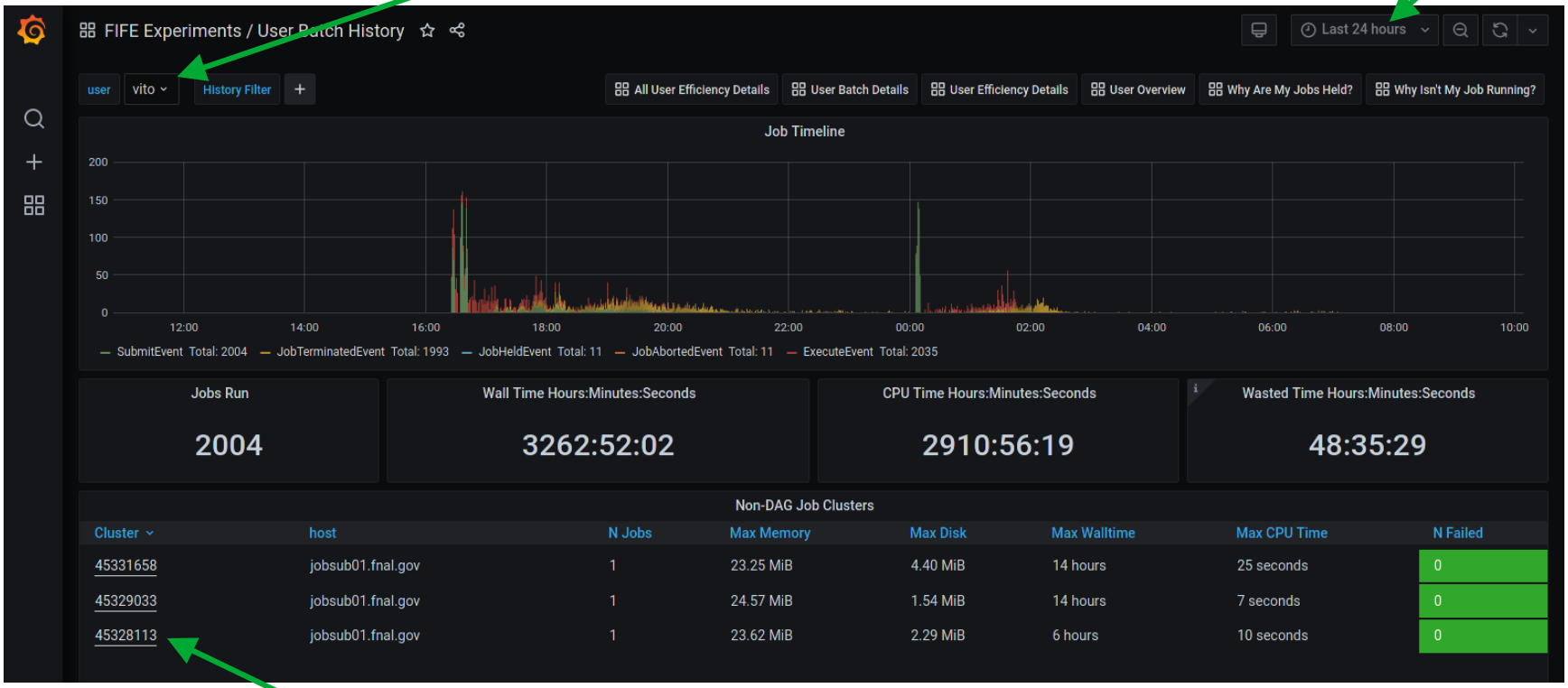
Monitoring jobs – Landscape

- User Batch Details



Monitoring jobs – Landscape

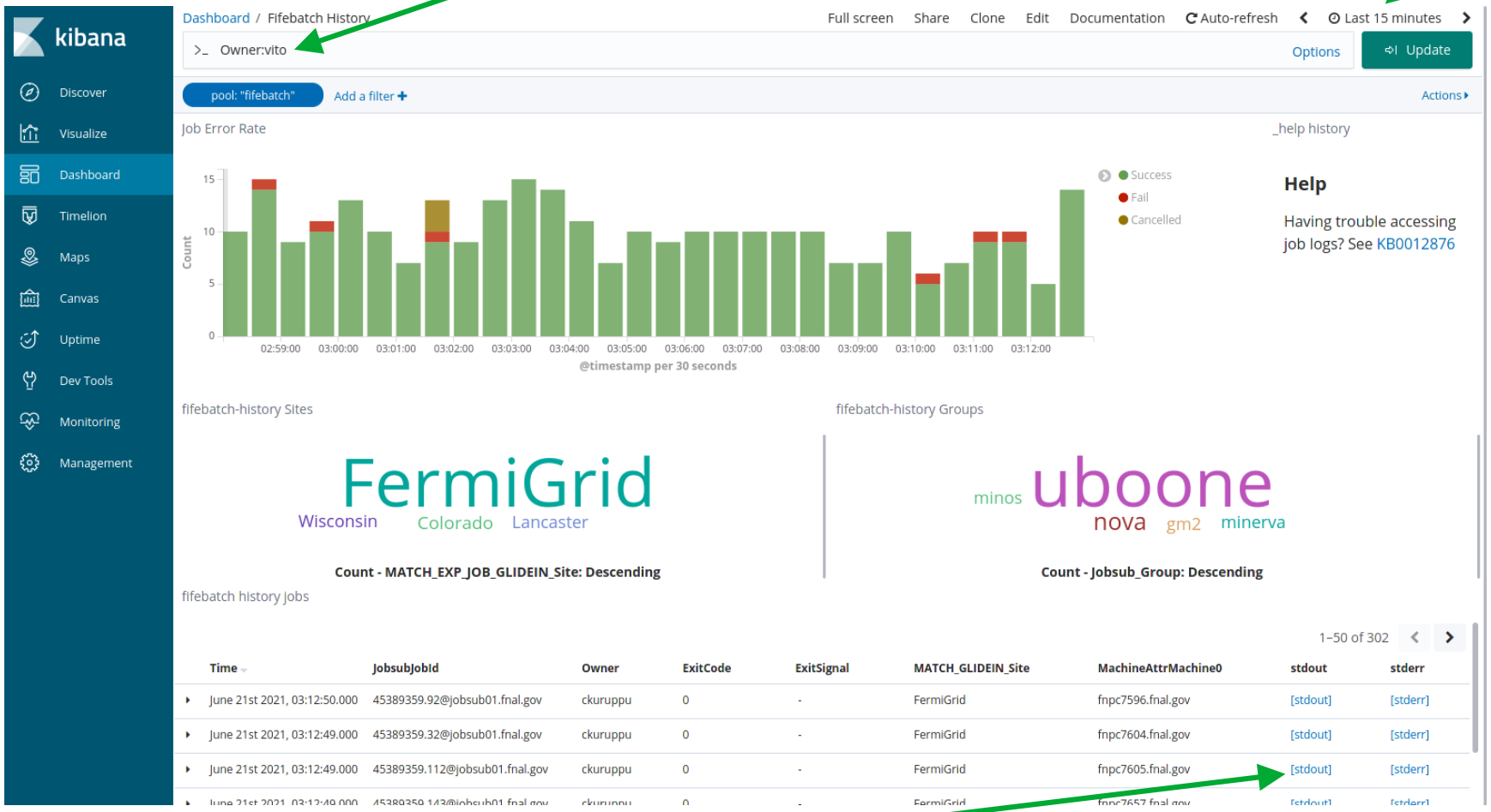
- User Batch History



Fermilab

Monitoring jobs – Kibana

- Kibana Fifebatch History



- Art Exit Codes

Summary

- Job submission to the grid is made through JobSub
- Experiments developed their submission tools based on JobSub
- Make sure to test new/updated workflow before submitting jobs at scale
- Set job requirements carefully to use resources efficiently
- Landscape allows jobs monitoring and provides access to jobs logs for debugging