# CLAS12 Software Environment, Computing Resources, Data Processing @ JLab

N. Baltzell

June 18, 2019

CLAS Collaboration Meeting

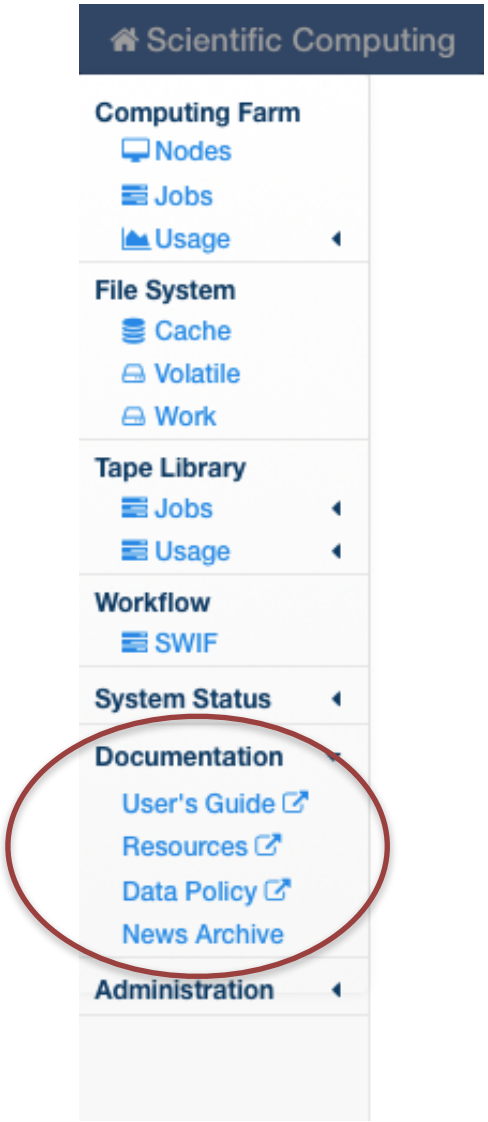
# General JLab scicomp Reminders

- See Bryan's scicomp talk from earlier today

- Batch farm PBS-SLURM Changes
  - Warning .... contrary to what was reported at the last collaboration meeting, SLURM now run your login shell again, with all its ~/.login ~/.cshc stuff!
  - if you don't specify stdout/stderr job tags, logs now go to
    - `/farm_out/$USER` instead of `$HOME/.farm_out`
    - this avoids quota issues in your `$HOME`, but you should still clean up your /farm_out

- Pay attention to emails from jlab-scicomp-briefs@jlab.org (click link for archives)
  - everyone with a JLab computing account *should* receive them
  - planned outages, system changes, etc

- Learn and use scicomp's documentation and monitoring web pages for batch jobs, disk quotas, tape access
  - http://scicomp.jlab.org

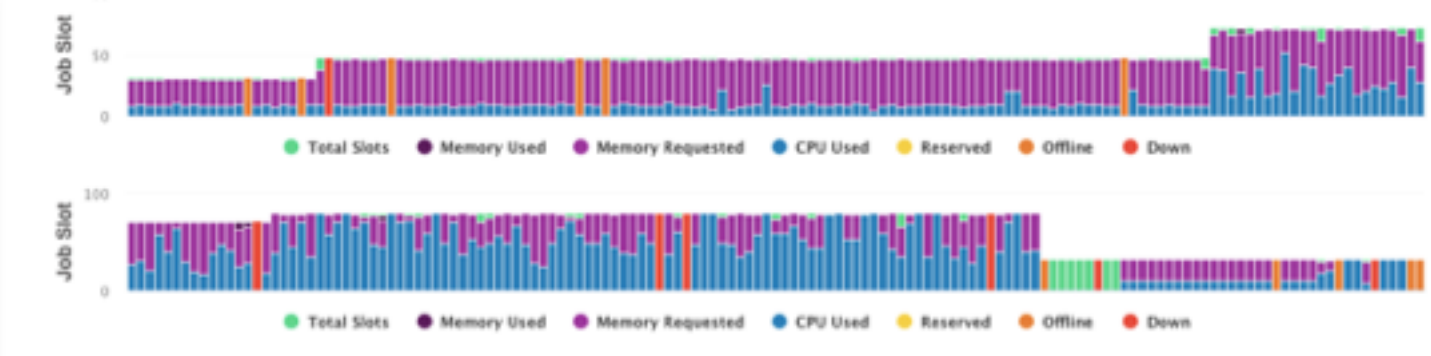


# Batch Farm Resource Requests (1)

Remember to optimize your requests, e.g. memory/cpu/disk/time. If you're running jobs on the Jlab batch farm, learn to use the metrics available here: https://scicomp.jlab.org.

Memroy request is particularly important for single-core jobs, due to the hardware of the Jlab batch farm which *averages* around 800 MB per job slot.

There will be a demo on Friday … (currently)



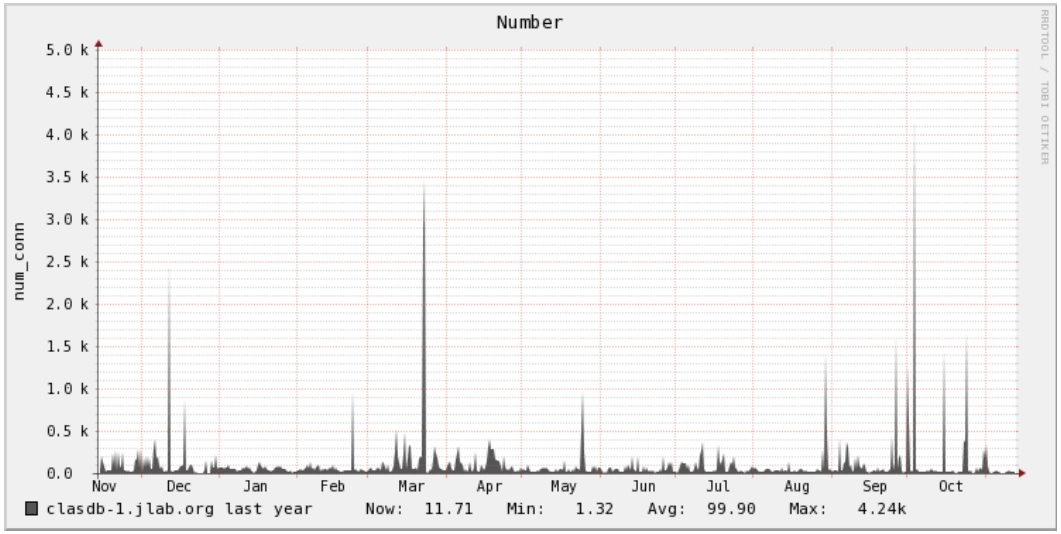

Jobs running on host farm140161

| JobId | project | User | JobName | Core | OS | MemReq | State | Submit | Start |
|---|---|---|---|---|---|---|---|---|---|
| 5377958 | casa | morozov | FarmJob | 1 | centos7 | 4 GB | ACTIVE | Oct-08 12:21 | Oct-08 12:32 |
| 5377960 | casa | morozov | FarmJob | 1 | centos7 | 4 GB | ACTIVE | Oct-08 12:21 | Oct-08 12:32 |
| 5377969 | casa | morozov | FarmJob | 1 | centos7 | 4 GB | ACTIVE | Oct-08 12:21 | Oct-08 12:32 |
| 5377970 | casa | morozov | FarmJob | 1 | centos7 | 4 GB | ACTIVE | Oct-08 12:21 | Oct-08 12:32 |
| 5400462 | PRad | xbai | InelEp_2GeV | 1 | centos7 | 4 GB | ACTIVE | Oct-09 05:11 | Oct-09 05:30 |
| 5400593 | PRad | xbai | InelEp_2GeV | 1 | centos7 | 4 GB | ACTIVE | Oct-09 05:17 | Oct-09 05:53 |
| 5400889 | PRad | xbai | InelEp_2GeV | 1 | centos7 | 4 GB | ACTIVE | Oct-09 05:32 | Oct-09 06:33 |
| 5401137 | clas12 | silvia | ndvcs | 1 | centos7 | 1 GB | ACTIVE | Oct-09 05:52 | Oct-09 05:54 |

Go to page: 1 Show rows: 10 1-8 of 8

# clasdb 2019+

- We've occaisonally had overloading issues with clasdb, which can lead to various services being offline, and batch farm jobs failing en masse
- clasdb is 20 years of accumulation of various databases, web services, shift schedules, service work, old logbooks, ... even some non-HallB stuff (which should be removed!)
- Plus CCDB/RCDB for clas12, where the software implements database access that is pretty well optimized (one bug to address)
- JLab IT recently provided a new set of servers, read-only, in-sync, on-site only, provisioned to handle all our batch farm load
- Onsite CLAS12 users should switch to the new server
  - Done automatically with the environment setup in this talk, or see manual documentation at the software wiki under CCDB



# CLAS12 Disk Storage @ JLab

**The large fileservers we commonly use:**

- `/work/clas12`
  - **175 TB, *manually* managed**
  - more traditional fileserver
    - good for lots of small files, small I/O operations
    - *not good for large data and large-scale I/O access (e.g. many simultaneous jobs reading lots of GB from the batch farm)*
- `/volatile/clas12`
  - **50/25 TB High/Guaranteed**
  - *automatically* managed ... pros and cons
  - Lustre, distributed system, good for large data I/O from the batch farm
  - ***scicomp is in the process of ~tripling Lustre, we can think about how to best distribute that between /cache and /volatile***

| Project Name | High Quota (GB) | Guarantee (GB) | Used (GB) | Small File(MB) | SmallFileCount |
|---|---|---|---|---|---|
| halla | 74,450 | 30,250 | 19,444 | 0 | 125,803 |
| hallb | 16,500 | 5,600 | 13,495 | 0 | 468,469 |
| hallc | 50,750 | 15,350 | 19,490 | 160 | 247,095 |
| clas | 61,000 | 21,150 | 24,808 | 0 | 197,161 |
| clas12 | 50,000 | 25,000 | 53,309 | 1,733 | 397,590 |
| cteqX | 500 | 200 | 0 | 0 | 0 |
| eic | 5,000 | 100 | 151 | 0 | 303 |
| halld | 60,000 | 30,000 | 54,848 | 4,541 | 2,112,717 |
| positron | 1,000 | 50 | 0 | 0 | 0 |
| | **319,200** | **127,700** | **185,545** | **6,434** | **3,549,138** |

**Both are always full. See [scicomp.jlab.org](scicomp.jlab.org).**

And we also adapted some tools from Hall D to better understand our disk usage, targets for improvement, etc. Here's the auto-deletion queue for /volatile:

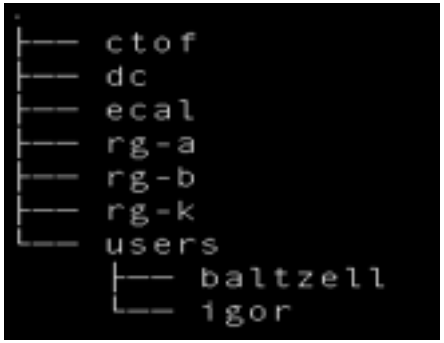
**http://clasweb/clas12offline/disk/volatile**

And breakdown on /work by directory, by user, file size/ age, etc:

**http://clasweb/clas12offline/disk/work**

**See demo on Friday ....**

We'd like to move to a more organized structure, with only run groups and detectors at the top level, and users inside a subdirectory:




- this will also enable moving towards finer-grained quotas, e.g. per run-group, separately from users easier, if we want to go that route
- Remember /work is not meant for permanent data storage, that's what tape is for!

# CLAS12 Software Environment @ JLab

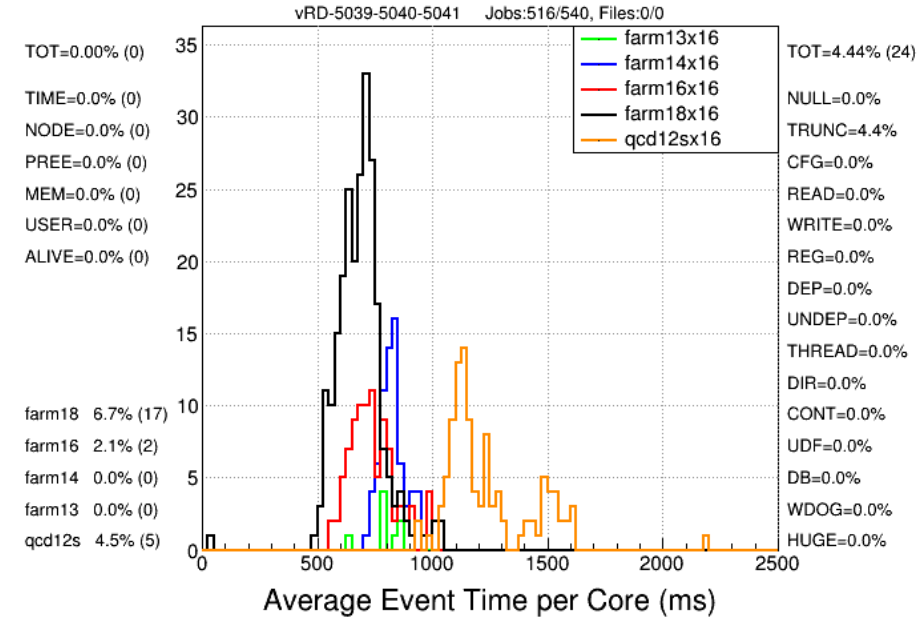
- A shared, group install of all clas12 software is officially maintained on the /group disk
  - To use it, you need to use "modulefiles" environment setup, but it's easy:
  - Source one file:
    - `source /group/clas12/packages/setup.csh` (or `setup.sh` if you use bash shell)
  - Then use the module command to see what's available, load them into your environment, see screenshot below.
- Rungroup chefs have been doing all data processing from these installs
- See demo on Friday, and documentation on the "uber" module, versioning, dev installs, etc, at the software wiki:
  - https://clasweb.jlab.org/wiki/index.php/CLAS12_Software_Center#tab=FAQ

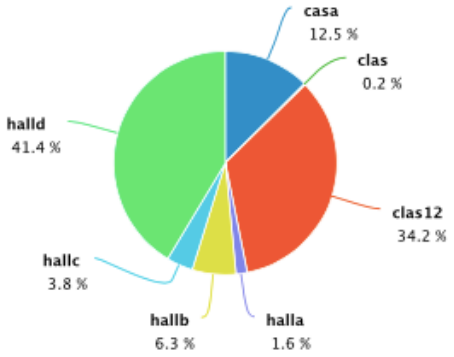
```
ifarm1801> source /group/clas12/packages/setup.csh
ifarm1801>
ifarm1801> module avail
ifarm1801> module avail

------------------------------------------ /group/clas12/packages/local/etc/modulefiles -------------------
ccdb/1.06.02            clas12root/dev          coatjava/dev            hipo/dev                root/6.12.06
ced/1.006e              cmake/3.15.2            evio/5.1                jaw/0.9                 root/6.14.04
ced/1.4.03              coatjava/6.3.1          gemc/4.3.0              jaw/2.0                 SubMit/dev
clas12/1.0              coatjava/6.3.1_4.3.11c  gemc/4.3.1              jdk/11.0.2              workflow/0.2
clas12/2.0              coatjava/6b.2.0         gemc/dev                jdk/1.8.0_31(default)   workflow/dev
clas12/dev              coatjava/6b.3.2         groovy/2.4.9            lz4/1.7.6
clas12/pro              coatjava/6c.3.3         groovy/2.5.6            maven/3.5.0
clas12root/1.0          coatjava/6c.3.4_4.3.11d hipo/1.0                rcdb/1.0

dulifarm1801> module load clas12/pro
ifarm1801> which hipo-utils
/group/clas12/packages/coatjava/6.3.1/bin/hipo-utils
```

# CLAS12 Data Processing @ JLab

Currently Hall B's net priority is **45%** of batch farm

- see [scicomp.jlab.org](scicomp.jlab.org)
- we've been developing additional tools to aid in studying our throughput, optimize it, get the most out of the resources
- including leveraging scicomp tools, Clara log file analysis (included in clas12-workflow on the next slide), which has allowed really tracking progress, implementing fixes, and optimizing Clara/ SLURM job configurations, memory usage, I/O logistics, etc




| JLab Farm | | | | | | | | CLAS12 Node | | CLAS12 Farm | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| flavor | memory (GB) | slots | memory per slot (GB) | nodes | slots*nodes | node fraction | slot fraction | node rate (Hz) | slot event time (ms) | rate (kHz) | rate fraction | events per day (M) |
| qcd12s | 31 | 32 | 0.97 | 195 | 6240 | 0.44 | 0.29 | 36 | 889 | 7.0 | 0.25 | 607 |
| farm13 | 31 | 32 | 0.97 | 22 | 704 | 0.05 | 0.03 | 41 | 780 | 0.9 | 0.03 | 78 |
| farm14 | 31 | 48 | 0.65 | 98 | 4704 | 0.22 | 0.22 | 62 | 774 | 6.1 | 0.22 | 525 |
| farm16 | 62 | 72 | 0.86 | 40 | 2880 | 0.09 | 0.14 | 94 | 766 | 3.8 | 0.14 | 325 |
| farm18 | 92 | 80 | 1.15 | 84 | 6720 | 0.19 | 0.32 | 120 | 667 | 10.1 | 0.36 | 871 |
| Weighted Average | | | | | | | | 63.4 | 773 | | | |
| Sum-Total | | | | 439 | 21248 | | | | | 27.8 | | 2405 |
| Hall B Fairshare | | | | | | | | | | 10.0 | | 866 |

| Playground | |
|---|---|
| Billions of Events: | 0.5 |

| flavor | days | days @ Hall B fairshare |
|---|---|---|
| qcd12s | 0.8 | 2.3 |
| farm13 | 6.4 | 17.8 |
| farm14 | 1.0 | 2.6 |
| farm16 | 1.5 | 4.3 |
| farm18 | 0.6 | 1.6 |
| Net | 0.2 | 0.6 |

| Fairshares | |
|---|---|
| ENP | 0.90 |
| Hall B | 0.40 |
| CLAS12 | 0.50 |
| Product | 0.180 |

# CLAS12 Rungroup Workflows

Incorporates decoding and reconstruction (NEW!), and soon trains (2019), into standard setup for chefs to easily submit large scale data processing jobs. Provides one standard, "easy" interface, no chef-scripting, no file list generation, no extra scripts/filelists lying around, for all CLAS12 run-groups.

Implements job-job dependencies to run full chain optimally, single-threaded and multi-threaded jobs, including file integrity checks internal to jobs, automatic launching of downstream phases, leverages Auger staging, leverages Swif and its retries, to achieve ultimately 100% hands free success. Can push to clas12mon for long-term monitoring, via standard Swif JSON format. (Note Swif is also the JLab-supported future conduit to offsite data processing)

With these tools, we processed now ~1M decoding jobs for RG-A/B with only a handful of system glitches (fixed via CCPR) and about 20K clara reconstruction jobs, with extremely low human intervention and very high success rate.
https://github.com/baltzell/clas12-workflow

| workflow_name | jobs | succeeded | success | attempts | phase | dispatched | depend | active | update_date | current_date |
|---|---|---|---|---|---|---|---|---|---|---|
| rga-decode9_R3135x178_x1300 | 62672 | 57482 | 91.7% | 74868 | 161 | 83 | 83 | 0 | Jun 18,2019 16:38 | Jun 18,2019 16:45 |
| rga-decode8_R3191x65_x1300 | 65648 | 65642 | 100.0% | 88013 | 67 | 0 | 0 | 0 | May 11,2019 11:03 | May 11,2019 15:44 |
| rga-decode7_R3249x43_x2100 | 67203 | 67200 | 100.0% | 72843 | 45 | 0 | 0 | 0 | Apr 21,2019 02:28 | Apr 21,2019 10:40 |
| rga-decode6_R3466x34_x2100 | 67597 | 67597 | 100.0% | 83442 | 36 | 0 | 0 | 0 | Apr 10,2019 17:35 | Apr 10,2019 17:40 |
| rga-decode5_R3355x30_x2100 | 65162 | 65162 | 100.0% | 68044 | 32 | 0 | 0 | 0 | Mar 25,2019 09:16 | Mar 25,2019 09:20 |
| rga-decode4_R3501x30_x2100 | 67620 | 67620 | 100.0% | 68232 | 32 | 0 | 0 | 0 | Mar 13,2019 14:06 | Mar 13,2019 14:10 |
| rga-decode3_R3750x29_x2100 | 66583 | 66583 | 100.0% | 67803 | 31 | 0 | 0 | 0 | Mar 4,2019 10:32 | Mar 4,2019 10:40 |
| rga-decode2_R3464x28_x1300 | 66415 | 66415 | 100.0% | 66998 | 58 | 0 | 0 | 0 | Feb 24,2019 04:54 | Feb 24,2019 05:00 |
| rga-decode1_R3432x25_x1300 | 64076 | 64074 | 100.0% | 79290 | 52 | 0 | 0 | 0 | Feb 11,2019 15:03 | Feb 11,2019 17:42 |