JLab Farm: Overview & Tips and Tricks

Brad Sawatzky

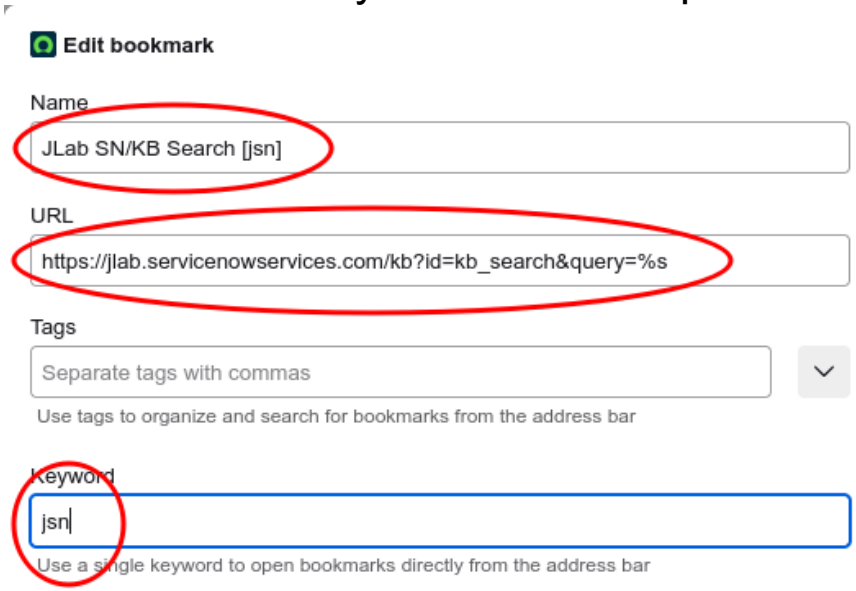
First up:
A Couple Quick Tricks to make your Computing Work Suck Less

How to find information

- JLab's web search and documentation kinda sucks…
  - → It is improving, but slower than I'd like … and with mixed results...
    - » Baby steps: ServiceNow Portal "Knowledge Base"
    - » Getting Started and Experimental Physics User's Guide pages are being updated
      - Searching is an ongoing issue ...
  - → Search trick: do this in Firefox:
    - » Go to www.google.com and search for this string: 'site:jlab.org OR site:jlab.servicenowservices.com foo'
    - » Right click on the bookmark and choose 'Properties'
      - Give it a good name
      - Give it a short 'keyword' like 'jj'
      - Clean up the URL as shown, replace 'foo' with %s
  - → Now type 'jj jget' in URL bar
    - » %s in 'Location' string is replaced with text following Keyword
    - » 'site:jlab.org' is google-fu to restrict search to jlab.org domain

How to find information

- Searching in JLab ServiceNow
  - → ServiceNow is where SciComp (and other groups) are putting their documentation.
  - → Search all of JLab ServiceNow from within Firefox:
    - » Go to https://jlab.servicenowservices.com/scicomp and login (top-right)
    - » Bookmark the page
    - » Right-click on the bookmark you made and update all 3 fields like so:



» Now you can type 'jsn <keywords>' in the Location bar for instant search

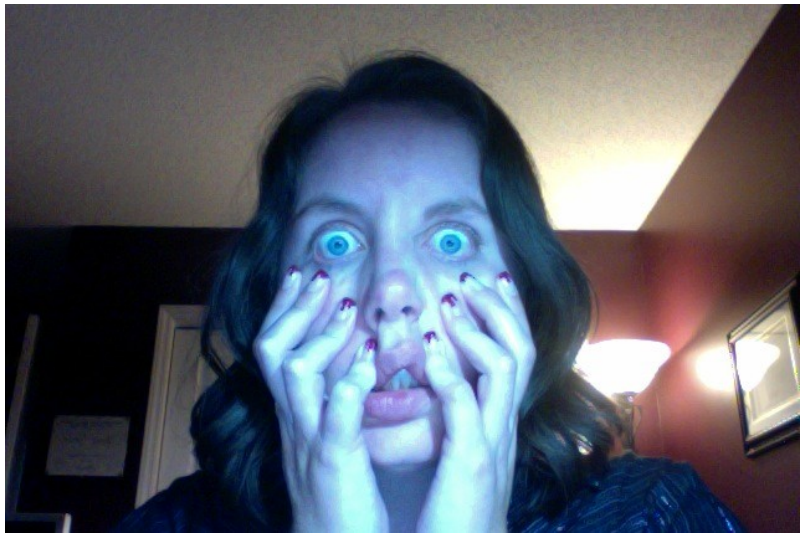
How to find information

- Trick works great for many things
  - →JLab staff page (https://misportal.jlab.org/mis/staff/staff.cfm)
    - » Keyword: 'jstaff'
    - » Location (can extract from search on 'smith' above):
    - » `https://misportal.jlab.org/staff_search?q=%s`
  - →ROOT / G4
    - » Keyword: 'gr'
    - » Location/URL:

      `https://www.google.com/search?hl=en&btnG=Search&q=site:cern.ch%20%s`
  - →Stackoverflow.com
  - →JLab Logbook (a little trickier, but you can work it out)
  - →...

How to work from Offsite

- How to work from offsite without tearing your eyes out because, holy hell, the graphics and menus are just so slow…



- Command-line (ssh) access
  - → Use 'ProxyJump'
    - » only 2-factor in once

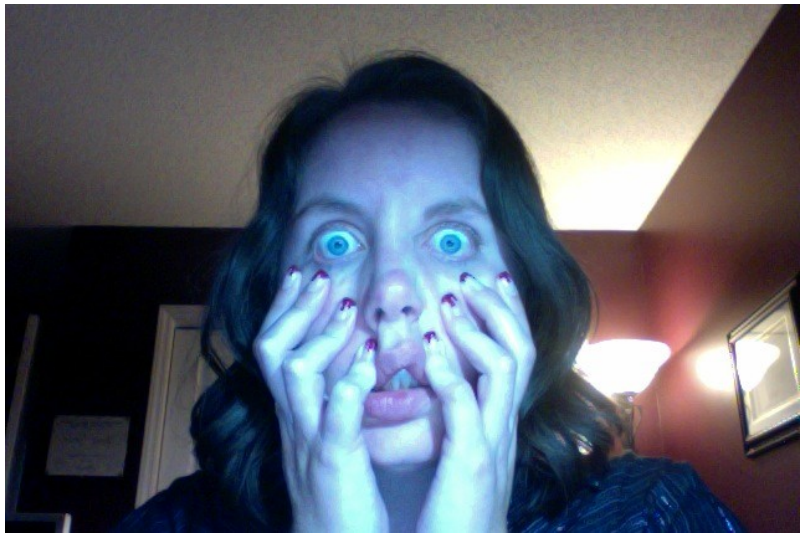
- VNC + ssh tunnel to the rescue
  - → VNC: Virtual Network Computing
  - → ssh tunnel is used to securely move VNC traffic through jlab firewall

- Old VNC 'howto' I wrote for my collaboration
  - → adapt to vncserver host you use (ie. jlabl2)
  - → Search: 'jj vnc session'
    - » Pick: Using a VNC Server/Client

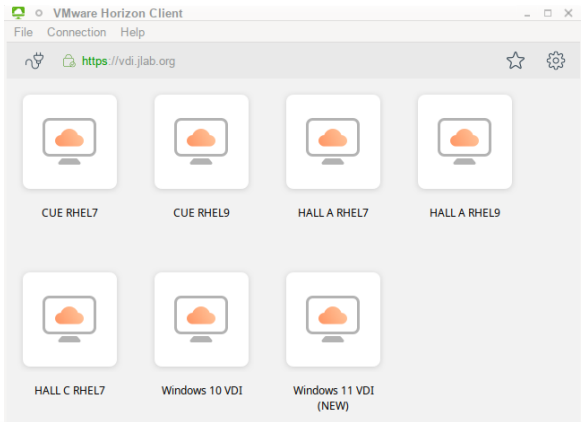
How to work from Offsite

- How to work from offsite without tearing your eyes out because, holy hell, the graphics and menus are just so slow...

- Virtual Desktop Infrastucture (VDI)
  - → https://vdi.jlab.org
    - » works within browser OR native application
    - » Windows 'just works', Linux requires a HelpDesk request
  - → Some Hall-specific options require you be granted access
    - » Compute Coord or HelpDesk
  - → Fewer "hoops" than VNC, but...
    - » limited number of 'slots' available
    - » sessions not as persistent
  - → NOTE: Turn OFF the auto-screenlocker in the remote session!
    - » (*Not* your desktop/laptop screenlocker though.)

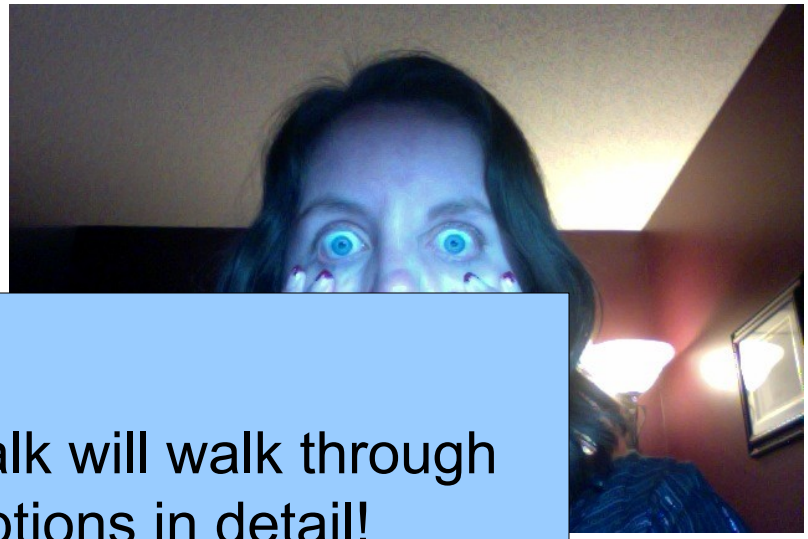


- Computer Center How-to
  - → Connecting using VDI



How to work from Offsite

- How to work from offsite without tearing your eyes out because, holy hell, the graphics and menus are just so slow...

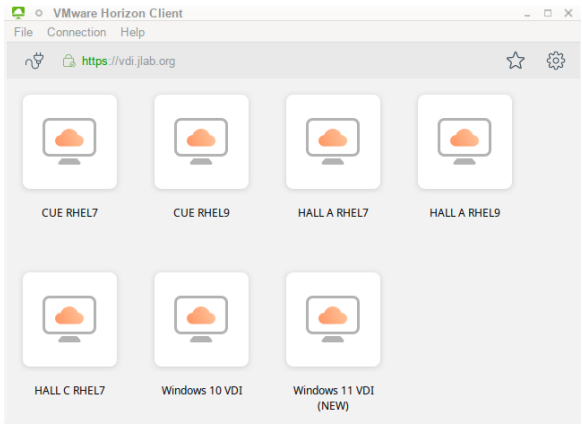


- Virtual Desktop Infrastucture (VDI)
  - → https://vdi.jlab.org
    - » wo
      ap
    - » Wi
      rec
  - → Some
    you be
    - » Co
  - → Fewer "hoops" than VNC, but...
    - » limited number of 'slots' available
    - » sessions not as persistent
  - → NOTE: Turn OFF the auto-screenlocker in the remote session!
    - » (*Not* your desktop/laptop screenlocker though.)

Cameron's upcoming talk will walk through several of these options in detail!

How-to

→ Connecting using VDI



JLab Email

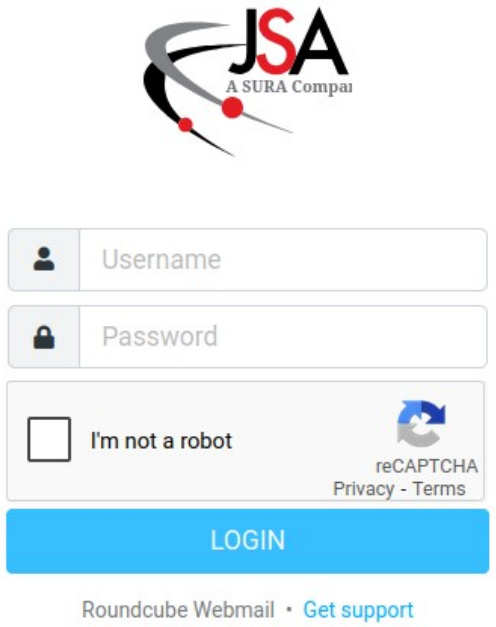
- Please monitor your @jlab.org address
  - →webmail.jlab.org
  - ~ OR ~
- Add JLab mail server to your preferred email client:
  - →Host settings
  - →Config Examples

(PSA: remember to update this when you update/reset your JLab password!)

» ~ OR ~

- Forward your JLab email to your 'main' account
  - →Helpdesk request

Watch for changes to user@jlab.org email to be announced Summer/Fall 2025!



Offline Analysis
Farm Usage / General JLab Computing

Nuts to the Farm, I analyze on my Desktop

- Simple tasks, some analysis OK on personal computer, BUT!!
  - → Thou shalt backup your code!
  - → Thou shalt backup your results!
  - → Who among us has “cleaned up”
    - % rm -rf stuff/
    - » Followed by !@#$?
- Don't keep only copies on your laptop
- Don't keep only copies on your desktop's hard drive
- Do use git for all code and scripts!
  - → Commit early, commit often
  - → 'git push' often too!
    - » It's a backup!

- Hard drives die and the data are gone.
  - → Drives are large and cheap
  - → But reliability on consumer drives is worse than it used to be!
  - → SSDs are (weirdly) no better!

- IF your hard drive died today, how long would it take to recover?
  - » a day,
  - » a week,
  - » a month???

JLab Systems can help!

- /home, /group are automatically backed up
  - →They are snapshotted hourly!

```
% cd .snapshot/
% ls -lrt
```

  - →Longer term backups are on tape

- /work, /volatile are on heavily redundant filesystems
  - →NOT backed up
    - » Use tape
  - →More on this later...

- NOTE: Your JLab RHEL, Windows, Mac system *can* mount these directories if needed
  - →Talk to me if this would help

The JLab Farm • Power at your Fingertips

- Farm has many components
  - → ~30000 compute cores
  - → ~11 PB Lustre
  - → ~5 PB NFS/XRootD (ZFS)
  - → ~100+ PB of Tape
  - → Consumes ~400kW!
  - → GPU nodes available too!
- Growth is $$$ and based on projections from Halls
  - → Expenditures generally switch between storage + CPU every other year

    (FY25 is 'Compute' year)

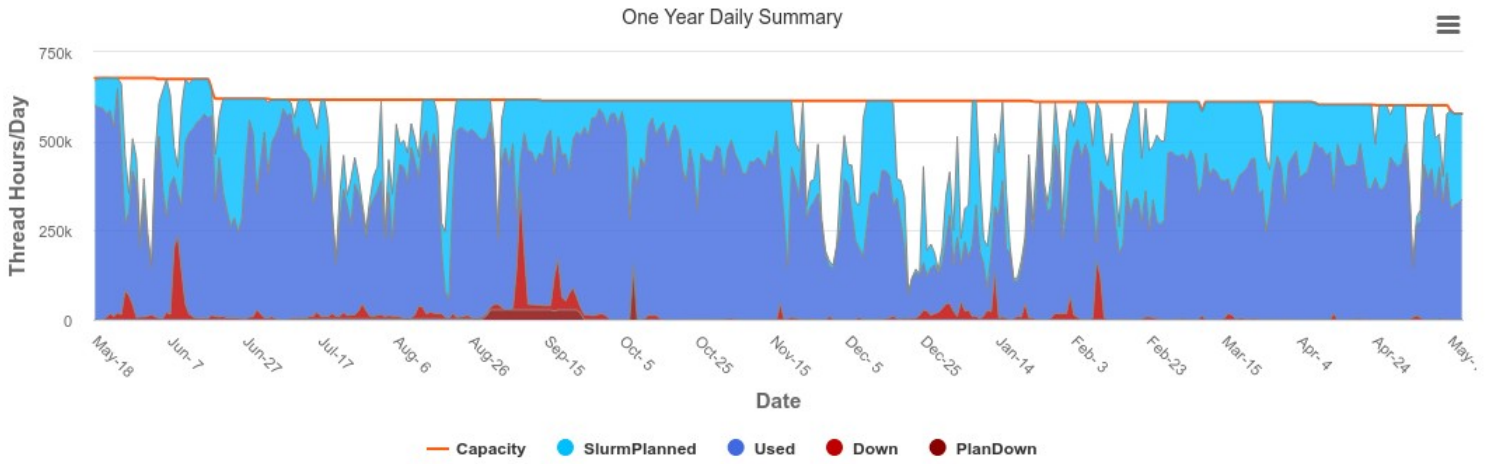
The JLab Farm • Batch Computing

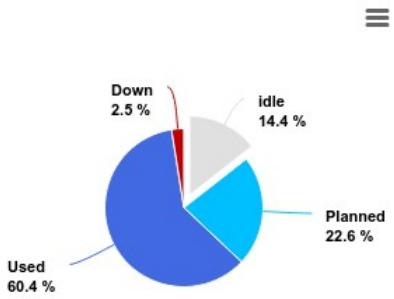
- The Farm: Batch Computing
  - → No direct access to these machines (*)
    - » Use "Interactive" farm nodes for testing
      - ie. ifarm, ifarm240[12]
  - → DB and other network access (git, http, etc) generally constrained
  - → Batch Jobs controlled by automated system called "slurm"
  - → You submit a job via slurm or swif and slurm schedules it to run
- All about trade offs:
  - → "Latency" can be high (hours+ from submission to job execution)
    - » BUT!
  - → Throughput is enormous
    - » 100s (1000s) of your jobs can run simultaneously
    - » High bandwidth access to fast storage
  - → A full replay (1000s of runs) can be completed in the time it would take a few runs to complete in series on your desktop/laptop.

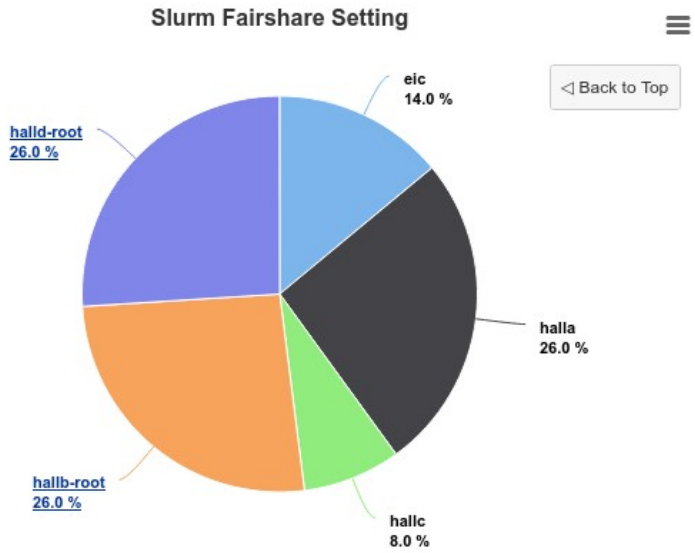
The JLab Farm • Scheduling

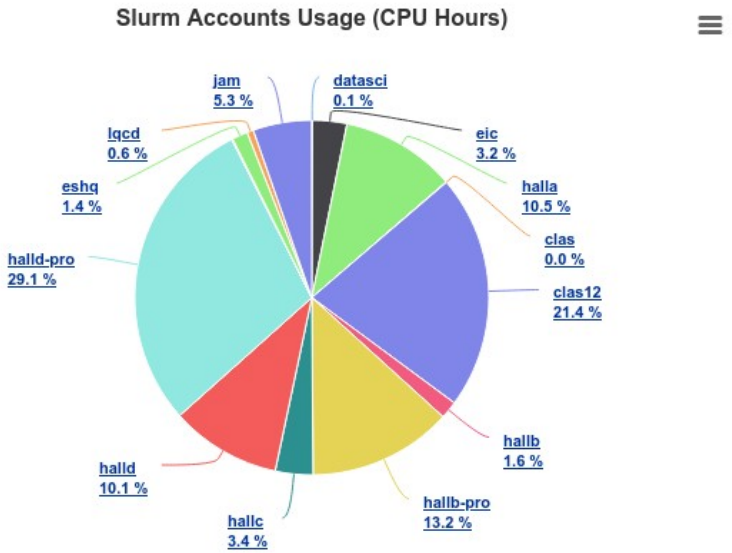
- The Farm is a Lab-wide shared resource
  - → Hall budgets include $$$ to support their workloads
  - → Rough allocation:
    - » A: 26%, C: 8%
    - » B: 26%, D: 26%
    - » EIC: 14%
- Ruled by Slurm workflow manager (*but you should use SWIF2!*)
  - → Allocations *not* written in stone and are adjusted based on needs
- The balance is trickier to manage than you may think...
  - → Jobs take time to run (system doesn't know how long beforehand)
  - → Upcoming job load is hard to predict
  - → System balances allocations over a few days, not hours
- More documentation here:
  - → https://scicomp.jlab.org/
  - → https://data.jlab.org/

The JLab Farm • Utilization









[Scicomp Farm Cluster Utilization]

How to Use JLab Batch Computing (Quick Overview)

Reminder: Do use the Farm!

- Note: The Farm is *not* your desktop
  - → It is a shared lab-wide resource
  - → Best to plan, test, and fire off groups of jobs
- Test your job first!
  - → Can it run reliably?
    - » If it doesn't run on ifarm, it won't run on the farm!
  - → Is the output what you want?
    - » Check before firing off 100 jobs

- Simple tasks, some types of analysis can be done on small systems, BUT!!
  - → Thou shalt back up your code!
  - → Thou shalt back up your results!
  - → IF your hard drive died today, how long would it take to recover?
- Don't keep only copies on your laptop
- Don't keep only copies on your desktop's hard drive

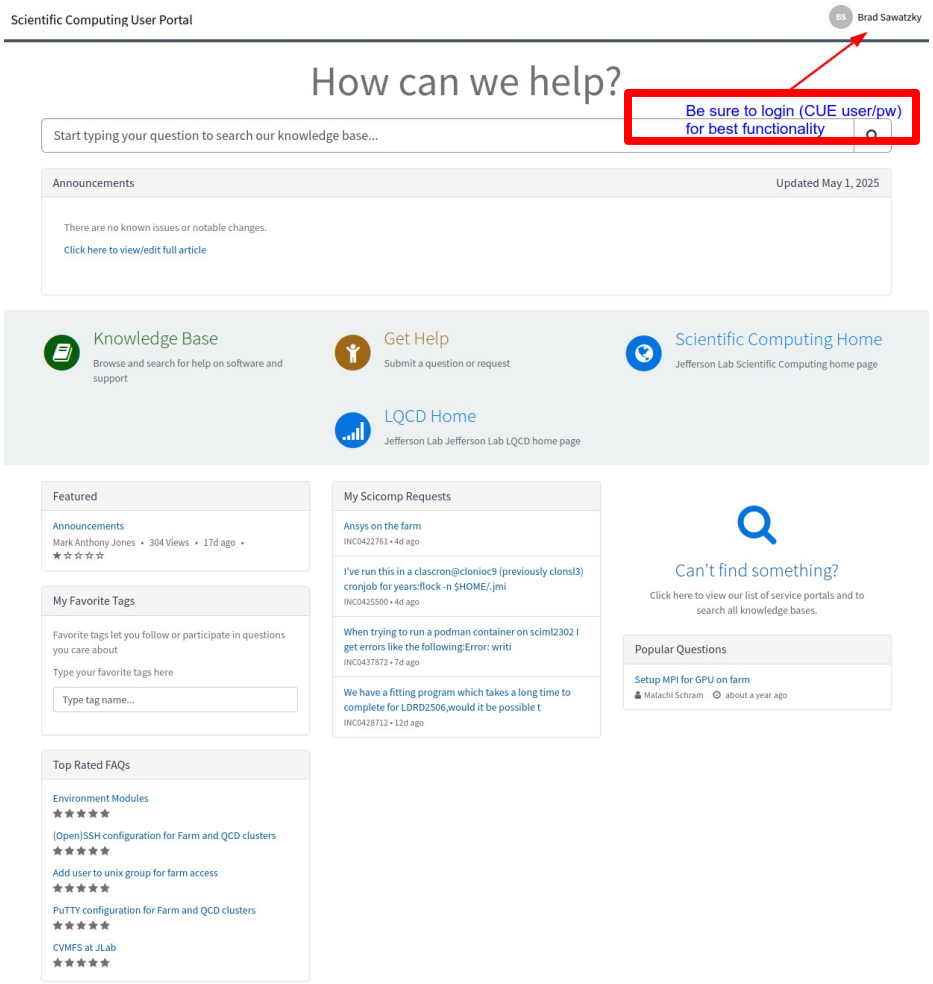
Basics: What's a “Job”?

- A 'Job' often maps to a shell script
  - → It can do multiple things, but usually it executes a single instance of your software
    - » Analyze one run, or
    - » Simulate “1M” events,
    - » *etc...*
- *NOTE: Output that would normally go to a terminal (ie. stdout/stderr) goes to special file system:*

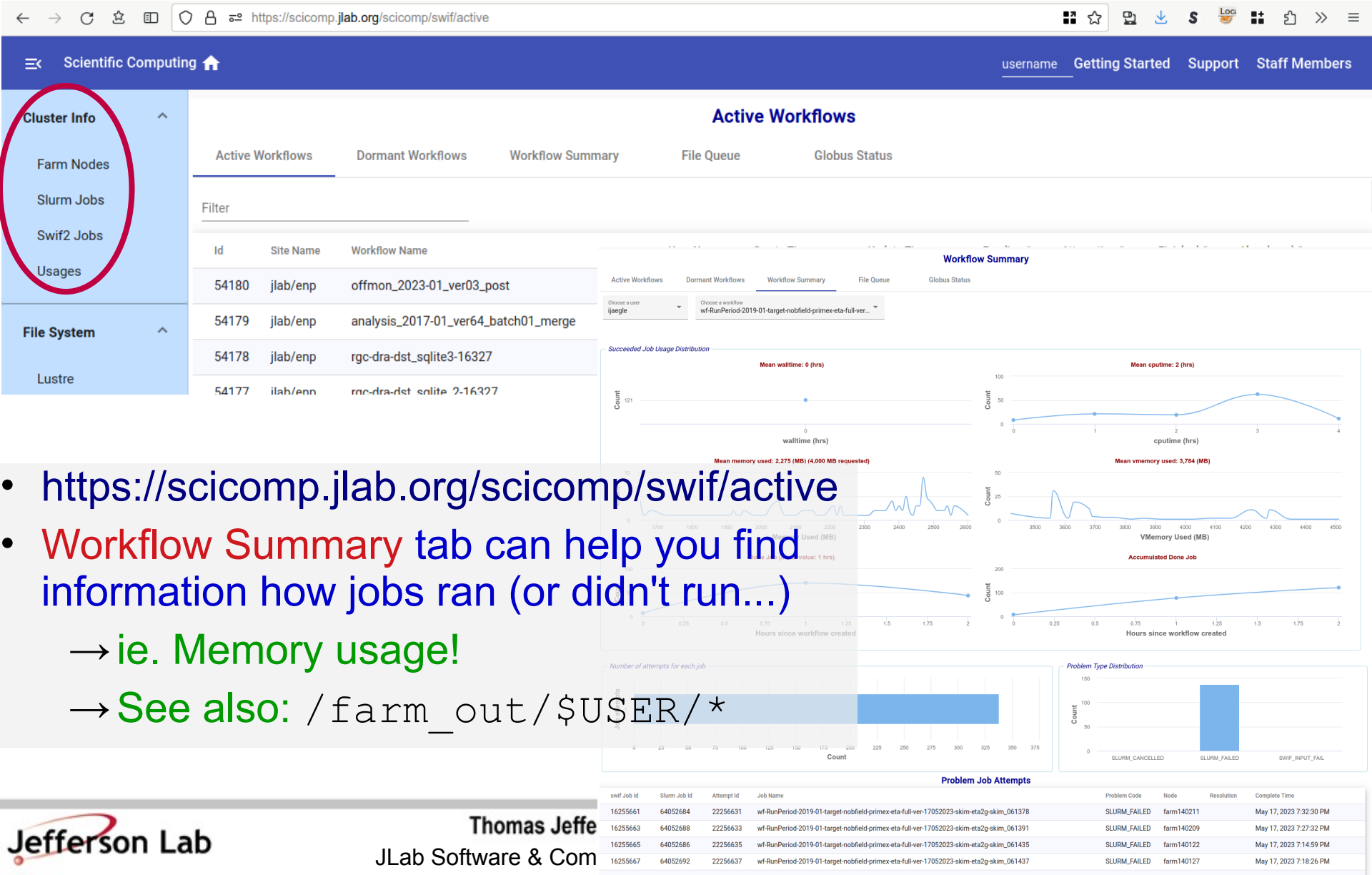
  /farm_out/$USER/job_id.out
  /farm_out/$USER/job_id.err

Help and Documentation moving to ServiceNow

→ https://jlab.servicenowservices.com/scicomp



Check Job Status



- https://scicomp.jlab.org/scicomp/swif/active
- Workflow Summary tab can help you find information how jobs ran (or didn't run...)
  - → ie. Memory usage!
  - → See also: `/farm_out/$USER/*`

Best Practices / Debugging a job

- Generally want a single script that does everything!
  - → Set up full environment
  - → Use full paths
    - » `/group/myExp/myscript.sh`
    - » ~~`./myscript.sh`~~
- Testing your script:
  - → 1st: Run on ifarm *and check*
  - → 2nd: Submit job to Farm
- Test with 'priority' 'partition'
  - → Max priority, fast sched.
  - → Limited 4 hour runtime
  - → Limited jobs/user
- Test on ifarm

```
% ssh you@ifarm
% /group/myExp/myscript.sh
```

  - → Make sure it worked!
    - » check histos, report files
- Quick Test on Farm

```
% swif2 add-job -create \
-partition 'priority' \
<other options> ... \
/group/myExp/myscript.sh
```

  - → Make sure it worked!
    - » check histos, files
    - » check `/farm_out/$USER/`
- Then submit full set!
  - → SWIF2!

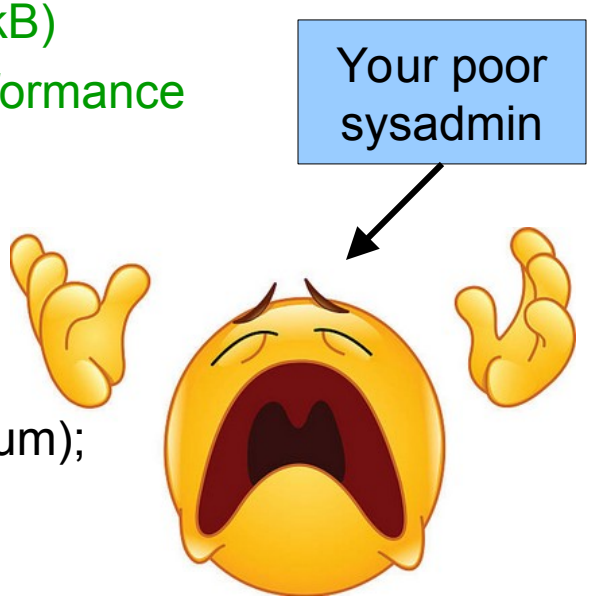
Swif/Slurm 'Debug' Commands

- How to debug a job failure on the Farm
- Note:
  - → "Job IDs" are not global
    - » SWIF job_id != SWIF job_attempt_id != slurm jid
  - → See Workflow Summary

- Find a failed SWIF job_id
  - → swif2 status -workflow <workflow> -user <user> -problems
- Look up failed job in swif:
  - → swif2 show-job -jid ####
  - → see info for each job attempt:
    - » *site_job_stdout*
    - » *site_job_stderr*
    - » slurm_id
    - » *job_attempt_problem*
    - » slurm_state
  - → seff <slurm_id>
- Use swif to rerun after fixes made:
  - → swif modify_jobs ...
  - → swif retry_jobs ...

Small I/O Problems

- Small read/write operations are *very* inefficient
  - → Old/legacy code defaults can be very small (~4kB)
  - → Should be closer to 4MB chunks for decent performance
  - → Buffered IO can bridge the gap if needed
    - » Common errors:
      - 'Debugging' output
        - » stderr << “got here” << endl;
        - » fprintf(stderr, “event %d\n”, eventNum);
      - Opening/closing files very frequently
      - Frequent random I/O
        - » ie. searching through a file for a parameter every event



- Workflows / procedures that may work on desktops or older systems do not scale well on modern systems (1000s of simultaneous jobs)
  - → Can take down / degrade system-wide filesystems
  - → Always be mindful you are on a large-scale shared system, not a personal desktop

Make your jobs schedule faster!

- Common Bottlenecks/ Mistakes
  - → CPU count
    - » ask for only 1 core only (unless you *know* the job will multi-thread!)
  - → Memory allocation
    - » < 2GB is optimal!
    - » Smaller → Faster scheduling!
    - » Test with '/bin/time -v <command>' on ifarm
    - » Look at SWIF workflow reports
  - → Insufficient debugging/ cross checks
    - » Fire off 100s of jobs with bad config, buggy code
- Request what you need, but use the resources you request.

Make your jobs schedule faster!

- Scheduling jobs takes many things into account
  - →File availability from tape
  - →Memory request
  - →CPU/core request
    - » >1 is often useless for common JLab codes
  - →'Fairshare' metric
    - » Average Hall utilization
    - » Hall Usage can be subdivided further
- Details
  - → Fairshare Web Page

- If a Hall / Project is not using 'their' fraction, then those Farm resources are available to anyone on a first-come, first-serve, basis!
  - →If the Farm is idle, you can take advantage!
    - » For example:

File Systems: Where do I put my stuff?

- SciComp/IT provides
  - →/home - your home dir; backed up by CST
  - →/group - a space for groups to put software and some files; system backed up by CST
    - » Like /home but for *groups*
  - →/volatile - acts as a scratch space for large files
  - →/work - unmanaged outside of quotas/reservations
  - →/mss - a 'directory' of what is on tape
  - →/cache - where tape files are written for active use

Where do I put my JLab stuff?

- /home/<you>/
  - →hourly snapshots
    - » `cd .snapshot/`
  - →personal, non-analysis files
    - » papers, notes, thesis, etc...
  - →analysis scripts: ~OK
    - » use git!
  - →source code: ~OK
    - » /work better
  - →NEVER store ROOT files or CODA files in /home or /group
- Your laptop / desktop
  - →Should really be just a front-end for working on JLab systems
  - →Everybody plans to do backups, but almost no one actually does backups until after they've lost data...

Where do I put my stuff?

- /group
  - → Think “/home” for work groups
    - » papers, thesis, etc
  - → hourly snapshots
    - » `cd .snapshot/`
  - → analysis scripts: YES
    - » also use git!
  - → source code: ~OK
    - » /work is better
  - → papers, thesis, etc in user subdirs is great
- /work
  - → Tuned for IOPs, small files
    - » ie. source, binaries, etc.
  - → NOT backed up
    - » but is resilient
    - » snapshots under .zfs/snapshot/ for *some* directories
    - » Do NOT count on this
  - → Source code: YES
    - » use git!
  - → ROOT/Simulation output:
    - » ~ick (don’t)
  - → CODA data: **No**
  - → YOU must backup to tape
    - » tar + jput *(more on this soon)*

Where do I put my stuff?

- /group
  - → Think "/home" for work groups
    - » papers, thesis, etc
  - → hourly snapshots
    - » `cd .snapshot/`
  - → analysis scripts: YES
    - » also use git!
  - → source code: ~OK
    - » /work is better
  - → papers, thesis, etc in user subdirs is great

- /work
  - → Tuned for IOPs, small files
    - » ie. source, binaries, etc.
  - → NOT backed up
    - » but is resilient
    - » snapshots under .zfs/snapshot/ for *some* directories
    - » Do NOT count on this

PSA: /work snapshots can be a pain because they count towards the quota for that space! (But you can't see them.)

- Generate big files, fill quota, whoops!
  - rm -rf <all the big files>
- quota still full!?!
- Talk to helpdesk… (nothing you can do)

Where do I put my stuff?

- /volatile
  - → Largest 'user' file system
    - » Petabyte scale
  - → High performance, tuned for large files
    - » ie. ROOT, simulation output
  - → NOT backed up
  - → Files auto-cleaned based on quota/ reservation/ and filesystem pressure
    - » https://scicomp.jlab.org/docs/volatile_disk_pool
    - » *Median file lifetime is >1 month*
  - → Analysis/Sim output goes here!
    - » Check results, then push to tape if good!
- Tape System
  - → Much bigger
    - » 100+ PB and growing
  - →/mss/hallX/...
    - » "Stubs": shows what is in the tape system!
    - » not the actual files
  - →/cache/hallX/...
    - » actual files
    - » auto-clean up in play
      - next slide

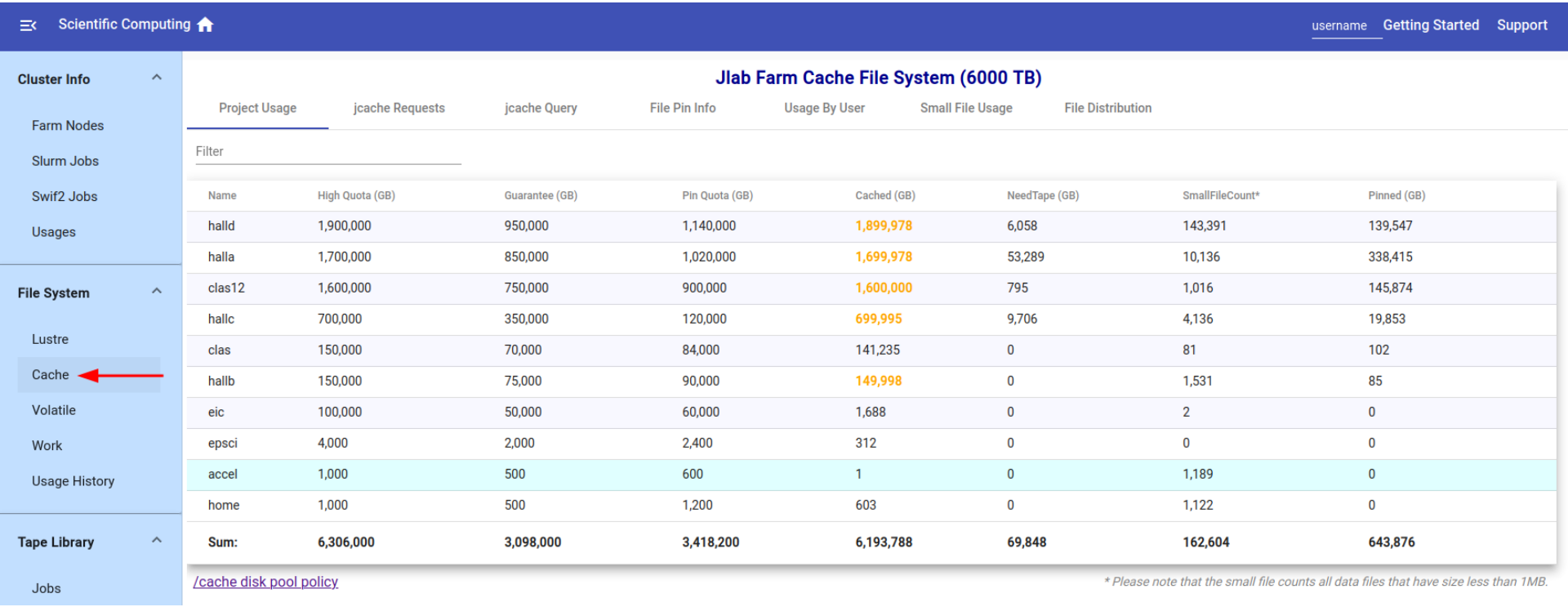
Accessing files from Tape

- Retrieving files from tape
  - →jcache get /mss/.../foo.dat
    - » Manual pull from tape to /cache/.../foo.dat
    - » Run 'jcache -h' on ifarm for documentation
    - » Never call this (or jget) in a farm script!
      - Let SWIF2 do it!
        - » List needed files as <Input> tag(s)
        - » Backend will pre-stage them for you in advance
    - » Please only manually pull the files you are going to use interactively.

      jcache get /mss/hallX/exp/raw/*

  - →jget /mss/.../foo.dat $PWD/
    - » pull file from tape to any CUE filesystem
    - » generally *not* the right tool

File duration in /cache

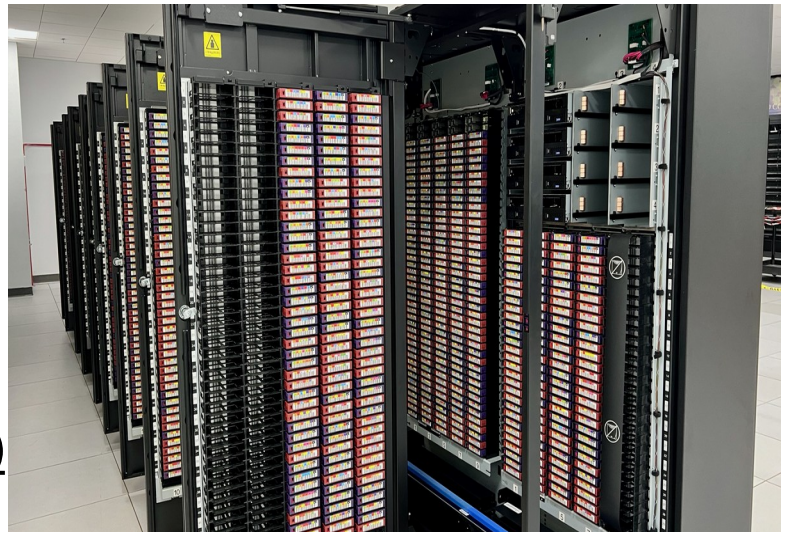


Name	High Quota (GB)	Guarantee (GB)	Pin Quota (GB)	Cached (GB)	NeedTape (GB)	SmallFileCount*	Pinned (GB)
halld	1,900,000	950,000	1,140,000	1,899,978	6,058	143,391	139,547
halla	1,700,000	850,000	1,020,000	1,699,978	53,289	10,136	338,415
clas12	1,600,000	750,000	900,000	1,600,000	795	1,016	145,874
hallc	700,000	350,000	120,000	699,995	9,706	4,136	19,853
clas	150,000	70,000	84,000	141,235	0	81	102
hallb	150,000	75,000	90,000	149,998	0	1,531	85
eic	100,000	50,000	60,000	1,688	0	2	0
epsci	4,000	2,000	2,400	312	0	0	0
accel	1,000	500	600	1	0	1,189	0
home	1,000	500	1,200	603	0	1,122	0
Sum:	**6,306,000**	**3,098,000**	**3,418,200**	**6,193,788**	**69,848**	**162,604**	**643,876**

- Files auto-cleaned based on quota and system pressure on /cache
  - → Clean up least-recently-used files first
  - → *Can* 'pin' files to keep them stable; but, *generally speaking, do not do this*
    - » If you do pin, you better be using the files interactively for the duration or you are literally getting in the way of your colleagues!
      - – For Farm jobs, use SWIF and declared inputs; the system will take care of it.
    - » /cache is a shared resource, be mindful of your impact on others!

Copying files to Tape

- Storing files on tape
  - → jput file /mss/.../
    - » 'jput -h' Online Docs
    - » ifarm2401% man jput
  - → jmirror dir local-prefix stub-prefix
    - » 'jmirror -h' Online Docs (Examples)
    - » ifarm2401% man jmirror
  - → swif2 add-job
    - -output <src> <mss://….>
    - » see swif2 documentation



- Note: JLab switched to read-only /cache in late Nov 2024
  - → *Used* to be able to write directly to /cache/…/…
  - → Might run across old scripts that attempt this, they will fail
  - → See Migration to read-only cache for details


WHAT'S NEW?
'ish

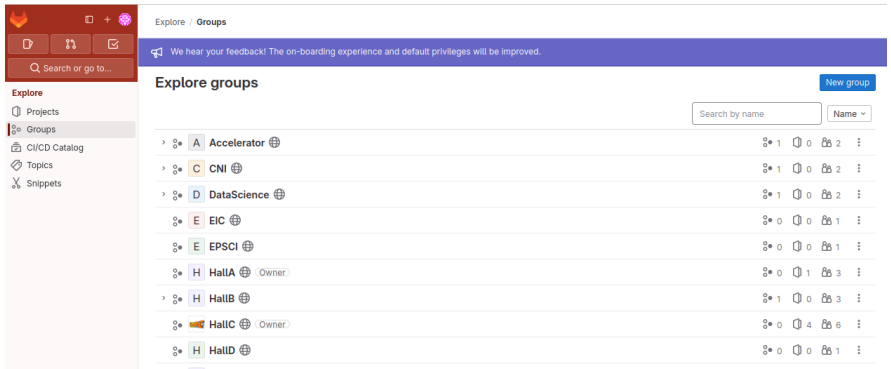
EL7 → Alma9 (Farm Transition)

- Farm OS fully running RHEL9
  CentOS7 (~RHEL7) →
  Alma9 (~RHEL9)
- (Much) newer default software, but be mindful of changes
  - → Update old farm jobs to remove 'el7' constraint
  - → must declare 'el9' constraint/ feature now
    - » Use:
      swif2 add-job -constraint el9 <other arguments>
    - » SWIF notes
    - » Slurm notes

- *Changes ...*
  - → */site, /apps* no longer mounted on farm nodes
    - » 'environment modules' framework (SW modules under /cvmfs, /group) instead
      - run 'modules avail'
    - » If something is missing, contact your Hall Compute Coordinator and/or open a Helpdesk ticket

code.jlab.org (GitLab Service)

- GitHub is getting $$$
  - → CI/CD, storage, *etc* are all metered costs
  - → JLab is on a 'legacy' license model for now but limitations are frustrating
- JeffersonLab GitHub Organization *will be maintained as-is*
  - → *BUT code.jlab.org should be a "value-added" proposition*
- code.jlab.org (GitLab instance)
  - → JLab run/managed
  - → Open / Offsite access
    - » Federated logins avail.
  - → CI/CD and Storage can leverage our Farm
  - → built-in Container Registry
  - → Supports several Data Management requirements for the Lab



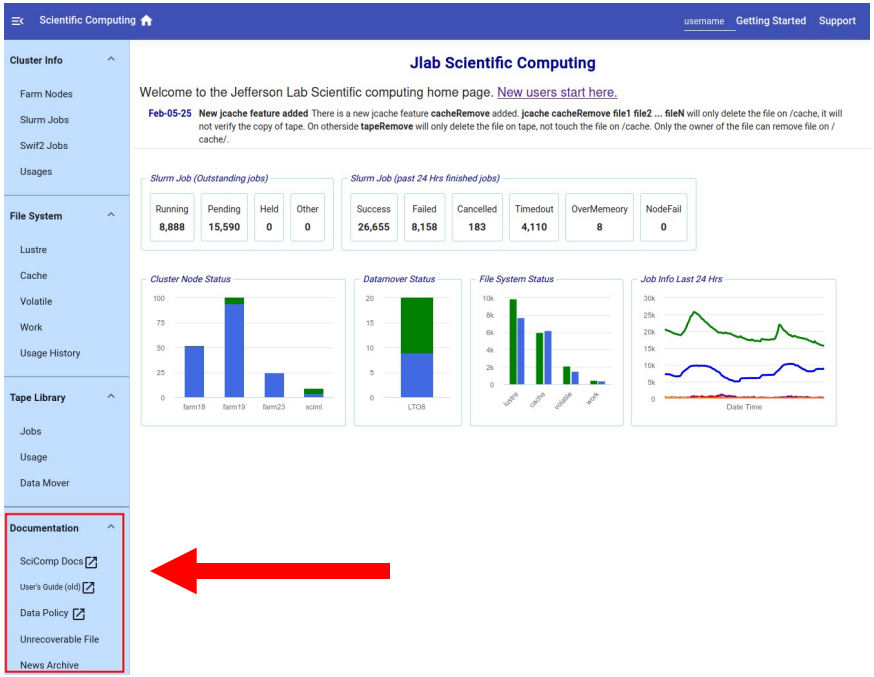
Containers: Podman / Apptainer

- Apptainer (was Singularity)
  - →works on both ifarm and farm
- Podman
  - →works on ifarm now
  - →should work on alma9 farm (some quirks)
- Docker
  - →not happening on compute clusters
  - →but podman == docker (pretty much)
  - →Note: docker != dockerhub

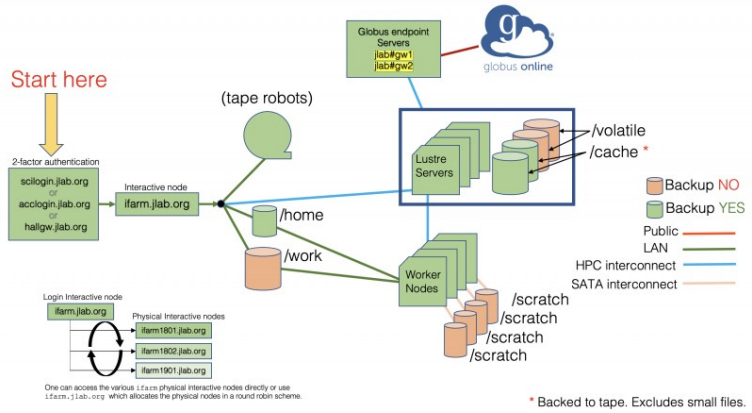
podman

Information Resources

- scicomp.jlab.org
  - →SciComp web page
- scicomp-briefs
  - →mailing list for JLab Scientific Computing



- Documentation links
  - →Getting Started
  - →SciComp Knowledge Base
  - →CST User Portal
  - →JLab Helpdesk
    - » helpdesk@jlab.org
    - » Incident Request

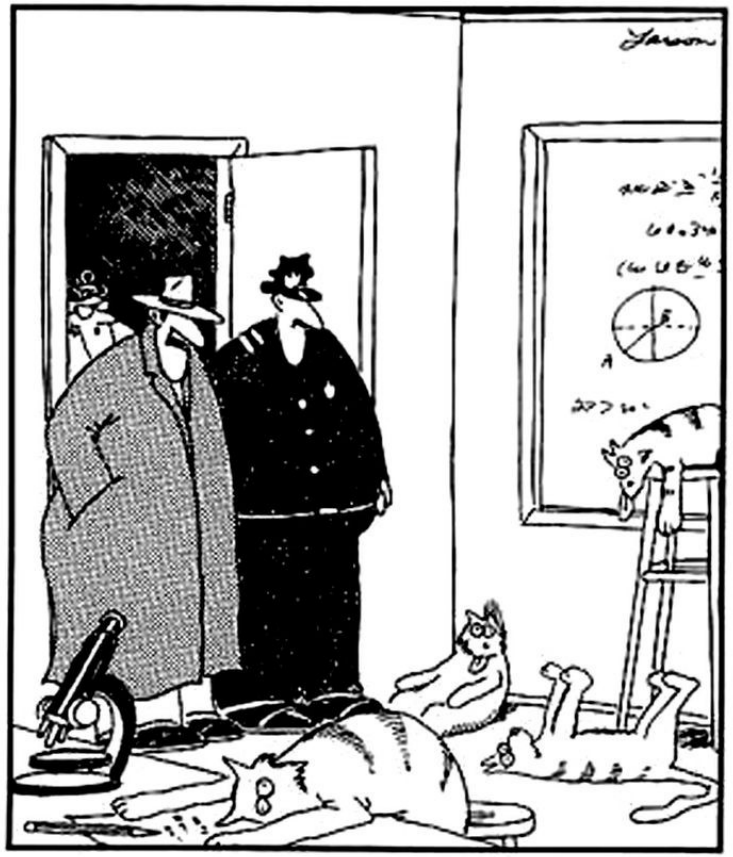


What do *you* need/want?

- Tell me what your challenges are!
  - →What resources are you missing?
    - » What are your bottlenecks?
  - →What applications/features do you want?
  - →Where do you / your collaborators struggle?
  - →IF SOMETHING IS REALLY ANNOYING, TALK TO US. WE CAN HELP!

- Feedback is necessary for SciComp / CST to plan
  - →(Also gives me “ammunition” to talk to management.)
  - →Email brads@jlab.org anytime

Now *Please* ask Questions!



"Notice all the computations, theoretical scribblings, and lab equipment, Norm. … Yes, curiosity killed these cats."