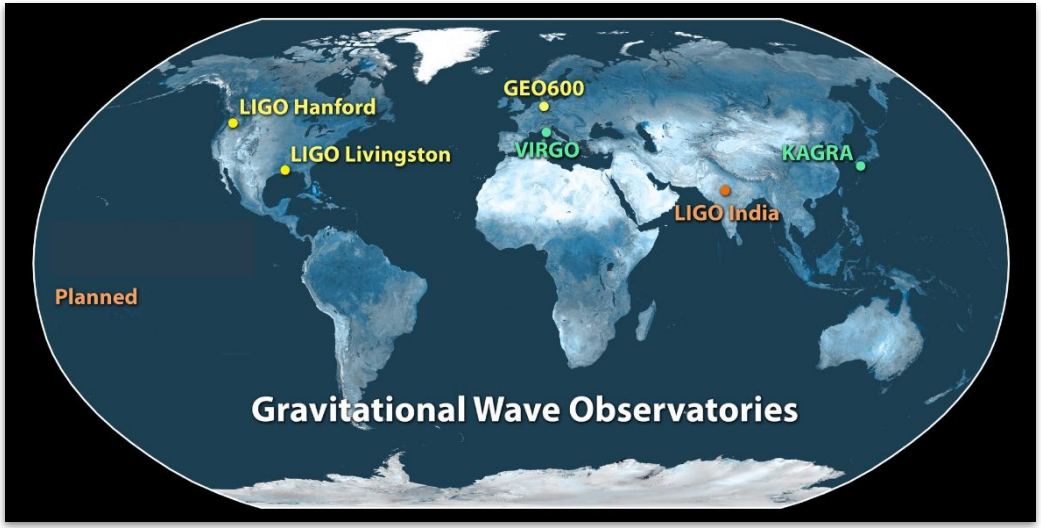
G2301299

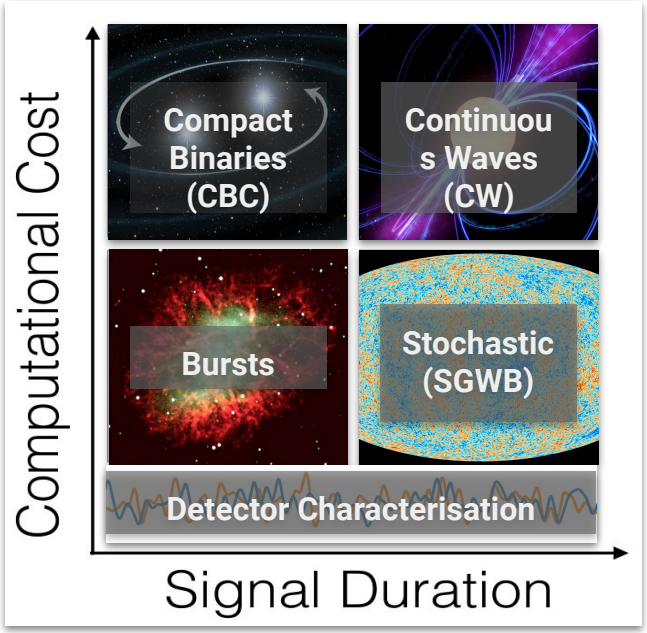
Exorcising the IGWN pool

Draining the swamp

james.clark@ligo.org, HTC23

IGWN: International Gravitational-Wave Network





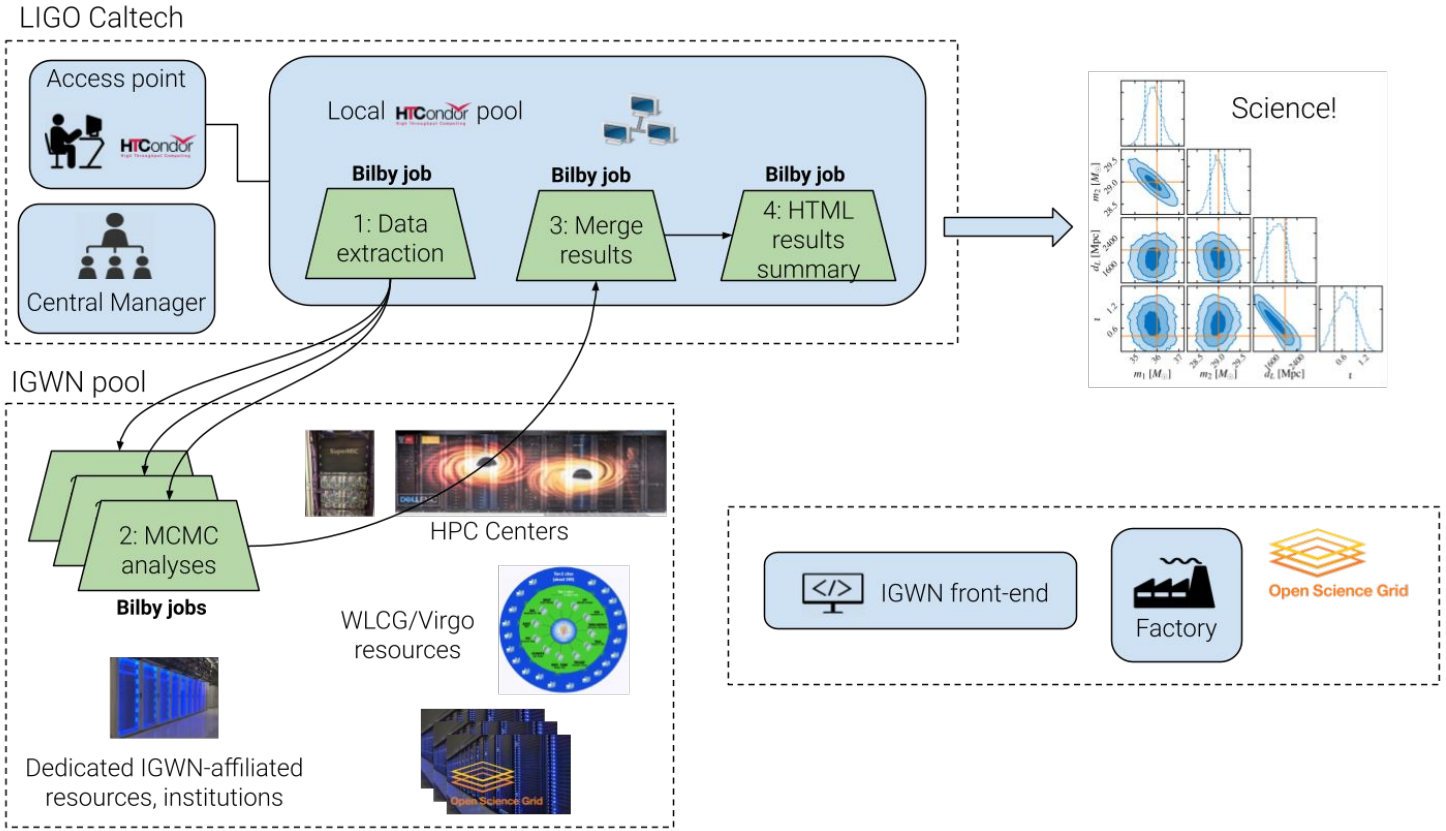
Astrophysics groups target different sources (see Keynote, Thurs)

Different sources & methods → zoo of different software, job, latency requirements, computational costs

Resource consumption still dominated by local submission to local HTC pools (local ~75% in last year)

Goal: move higher latency, CPU-expensive / GPU analyses to distributed HTC pool

The IGWN pool (and a representative analysis pipeline)



Much to keep track of

Overall functionality/plumbing:

- Communication across access points, collectors, entrypoints, frontend and factory
- Jobs running at all sites which support LIGO/Virgo VOs?

“List Of Doom”:

- Had a large number of “missing” sites where jobs should / had previously run
- Systematically worked through w. OSG to troubleshoot, test → mostly resolved

Job performance:

- Job success & goodput
- Data access

“List Of Woe”: documenting sites with suboptimal glidein configs (e.g., no multicore slots @ LIGO sites)

Testing / demo’ing new(ish) HTCondor/OSG functionality

Challenging to keep track

Often (historically) intermittent / stochastic science usage → lack of constant pressure

- Easy for site-level outages to go unnoticed
- Hard to distinguish large-scale problems from lack of demand

Small (but growing!) base of users in the IGWN pool

- Power users: often find workarounds (can't be trusted to report problems)
- Novice users: easily scared → fall back to dedicated resources & local pools

Nagios-style checks:

- Great for host statuses & service status (where accessible) [WIP]
- Less appropriate / harder to design for site- & application-specific problems

Need some way to "exercise" [G.Thain: "exorcise"] infrastructure and monitor *realistic* user experience

Introducing: “Grid Exerciser”

Periodic submission of a DAGMan workflow to test / profile:

- Availability / functionality of CPU & GPU resources
- IGWN data discovery
- IGWN proprietary data access via CVMFS / OSDF client file transfers
- Access to CVMFS-hosted software repositories
- condor_ssh_to_job functionality

Grid exerciser job histories → aggregated into elasticsearch by `condor_adstash` & presented on

- Grafana dashboard, grouped by site / application
- Daily email summary

DAGMan workflow also attempts to demo/test various HTCondor functionality

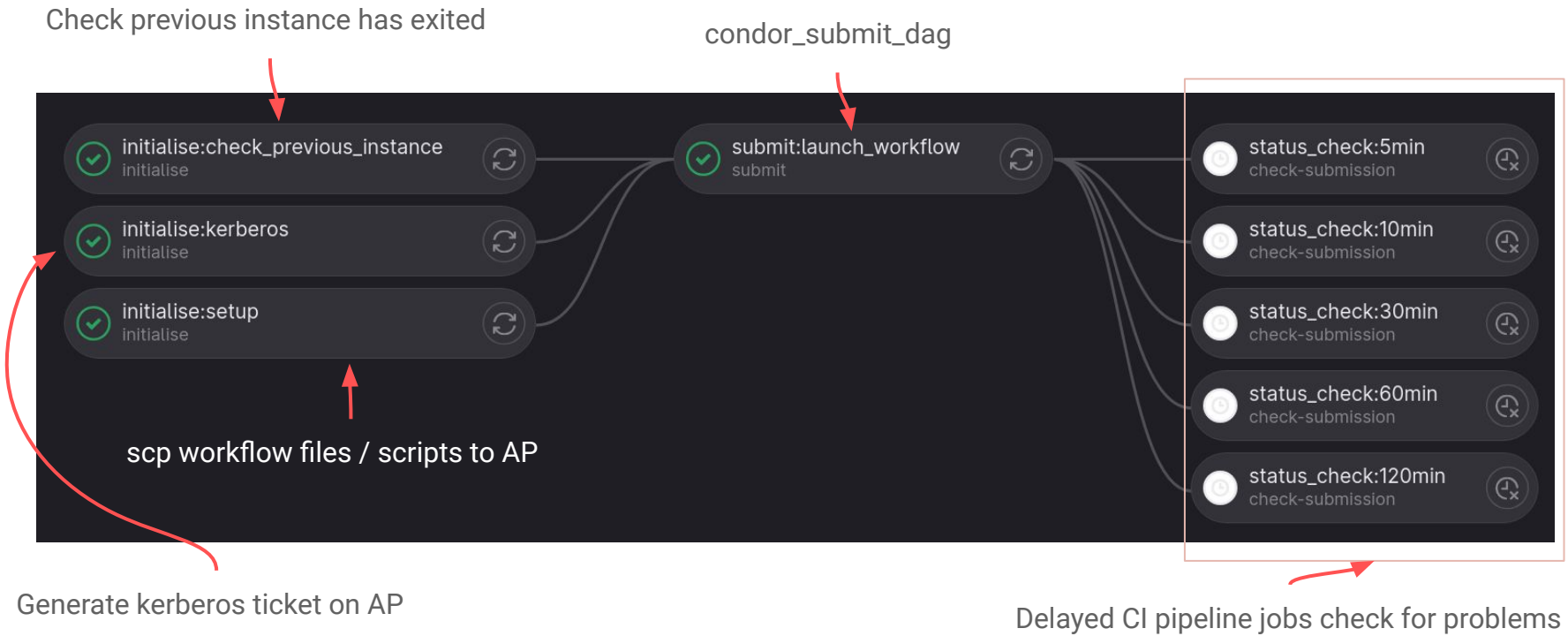
- Parent DAG (now) has a mix of `JOB`, `SERVICE`, `SPLICE`, `SUBDAG` and `FINAL` nodes
- DAG files for `SUBDAG` nodes generated on the fly as a parent job of the `SUBDAG`s (via python bindings)

The ropes and pulleys

3-hourly DAG submission via scheduled GitLab CI pipeline

CI pipeline failures → email alerts, easy visualisation

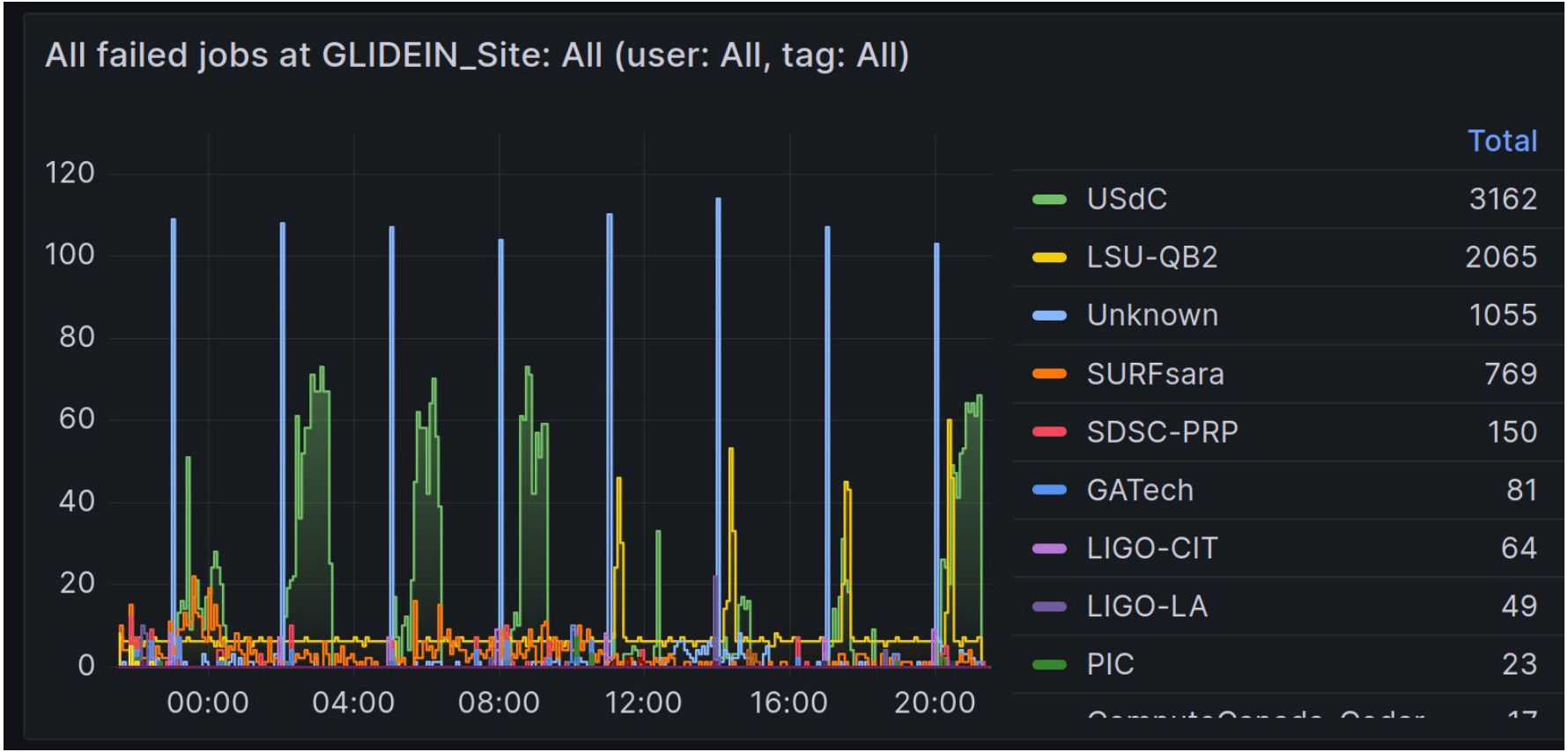
Easy to configure run “modes” (e.g., setup only, nosubmit)



IGWN pool dashboard: strategic overview

Site	# jobs	Job success rate	Average RemoteWallClockTime	Goodput
Wisconsin	334	100%	7 hour	22%
Vanderbilt	6243	100%	24 min	52%
Unknown	6064	83%	16 min	97%
USdC	5044	36%	2 hour	51%
UConn-HPC	593	100%	4 hour	55%
UChicago	2076	100%	12 min	56%
Swan	181	100%	22 s	100%

IGWN pool dashboard: where are any jobs failing?

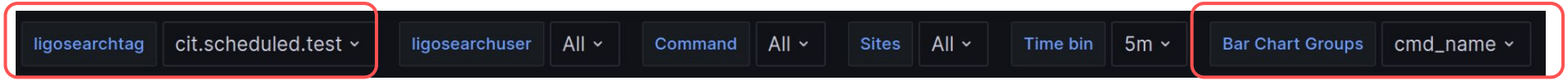


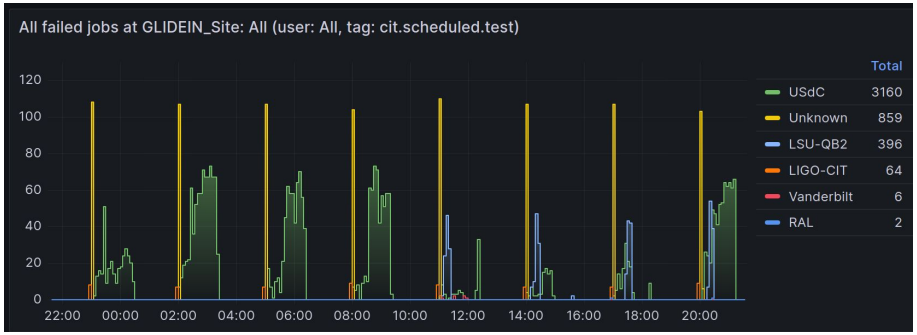
IGWN pool dashboard: whose jobs are failing?

Failures for past 24 hours: mostly grid-exerciser tests

Can drill down by fixing “ligosearchtag” and grouping by user / application…

IGWN pool dashboard: grid-exerciser view

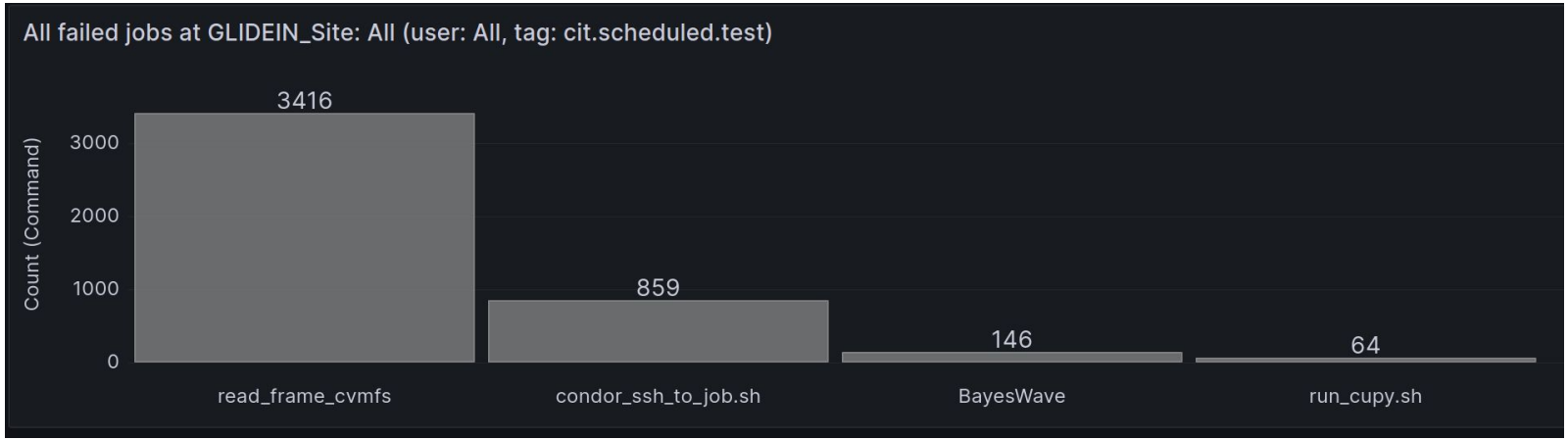




Filter down to grid-exerciser jobs

Configure bar chart for failing executables

Grid-exerciser failures always dominated by: auth. CVMFS & ssh-to-job

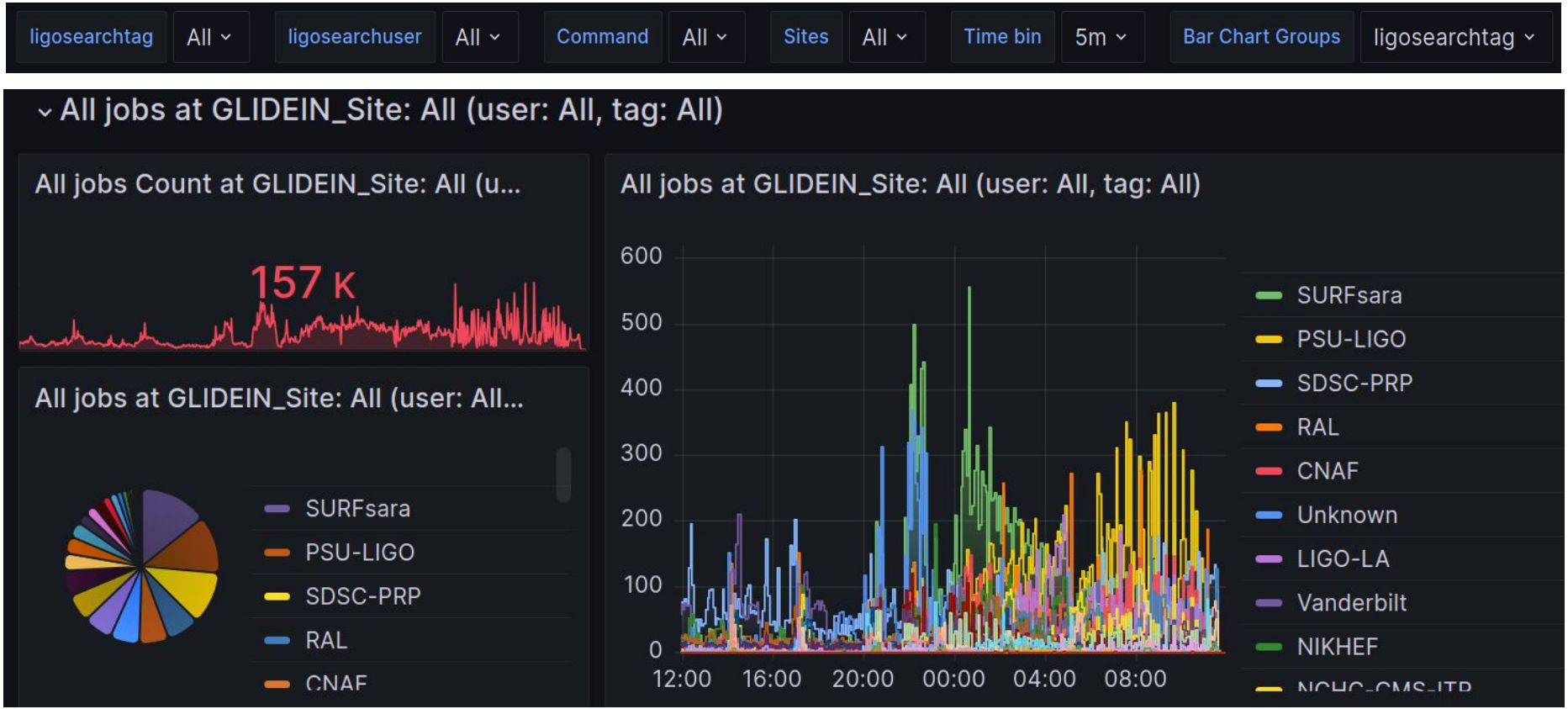


(Some) current problems / gripes…

1. ~~DAGMan started assertion errors (SERVICE nodes are broken)~~ [understood: HTCONDOR-1909]
    a. gitlab-CI check of previous instance *always* fails
    b. No DAGMan metrics file → condor job-triggered gitlab-CI pipelines *always* fail
2. No (?) meaningful measurement of goodput for self-checkpointing applications
3. OSDF client + condor file transfer failures → held jobs, I want to identify failures (~easy to fix my tests)
4. Many teething problems with transition to SciTokens (in payloads):
    a. SciTokens & `condor_submit`: 😀
    b. SciTokens & `condor_submit_dag`: 😣

Extras

IGWN pool dashboard: where are jobs running?



IGWN pool dashboard: condor_ssh_to_job

Want to identify sites where ssh-to-job is ok

Script: waits until other *target* jobs in the DAG enter run state.

Once target is running, local universe SERVICE node job:

- `condor_edit`s itself to record *target site*
- `condor_ssh_to_job <target jobid>`
- success / failure → elasticsearch & grafana

SERVICE nodes: "**typically used to run tasks that need to run alongside a DAGMan workflow**"

SERVICE node means DAG completion is independent of whether ssh-to-job ran

condor_ssh_to_job success rates

condor_ssh_to_job	# jobs	Success rate
undefined	200	78%
Wisconsin	622	40%
Vanderbilt	226	38%
USdC	292	46%
UConn-HPC	24	71%
UChicago	59	8%

Mixed success rates: ssh-to-job *had* been working ~well until about a week ago…