Monitoring and Analytics With HTCondor Data

William Strecker-Kellogg
RACF/SDCC @ BNL

RHIC/ATLAS Computing Facility (SDCC)

Who are we? See our last two site reports from the HEPiX conference for a good overview of who we are and what we do.

Recent news: running singularity for ATLAS jobs in production

In short: support computing for RHIC collider at Brookhaven National Lab

- Condor pools for RHIC, 2 x ~16k core + smaller experiments

In short: act as the Tier-1 for ATLAS in the US, storing data from CERN and distributing across US, as well as run large compute farm

- Host 90Pb of tape storage, dozens of Pb disk on Ceph, dCache, etc
- Runs around 18kCPU farm

Increasing HPC presence, GPU-based Cluster + new OPA + KNL cluster

Batch Monitoring

Batch Monitoring

Why?

1. Analytics

- Is my farm occupied?
- With whom?
- No jobs are starting! Send Alerts!
- Who's killing my storage!?

2. Accounting

- Who exactly ran, when, where?
- Let's charge them per cpu-hour
  - Anathema to HTC
- Exactly how full were our resources?
- How cpu-efficient were we?

Batch Monitoring

Why?

Batch Systems, especially HTCondor, are great at producing vast streams of low-meaning data

Wrangling it together to distill meaning / utility is a data-science challenge

Condor cares who has what work and where it can be run.

> Really, really good at this. But. Plops bits and pieces of state history all over in log files on the schedd and startd side

State?

Distributed. Schedds know jobs, startds know running jobs, periodically update schedd

Collector knows little; just enough to query and match.

Matching is stateless

Only persistent pool-wide metadata (excluding offline ads) is Accounting data

Kept in logfile where negotiator runs

Need for Accounting: Current Status

Command digests history log, reads "***"-separated classad text

Dumps to ever-growing mysql DB

Carefully watches job status transitions for restarts and evictions, generates statistics about efficiency per-job and good(bad)put, etc...

> This is ugly. I did this. Years ago. Don't look at it.

Doesn't this sound like a great case for logstash, Spark, etc... (insert trendy "big data" analytics framework here...)?

~~Accounting~~
Analytics

Background of Monitoring Technologies

- 3 Components: Collect, Store, Display
  - Display: Grafana
  - Collect: Fifemon + own code
  - Store: Graphite
- Additional aggregation / analysis layers
  - Statsd
  - Carbon-aggregator
  - Elasticsearch

Graphite Architecture

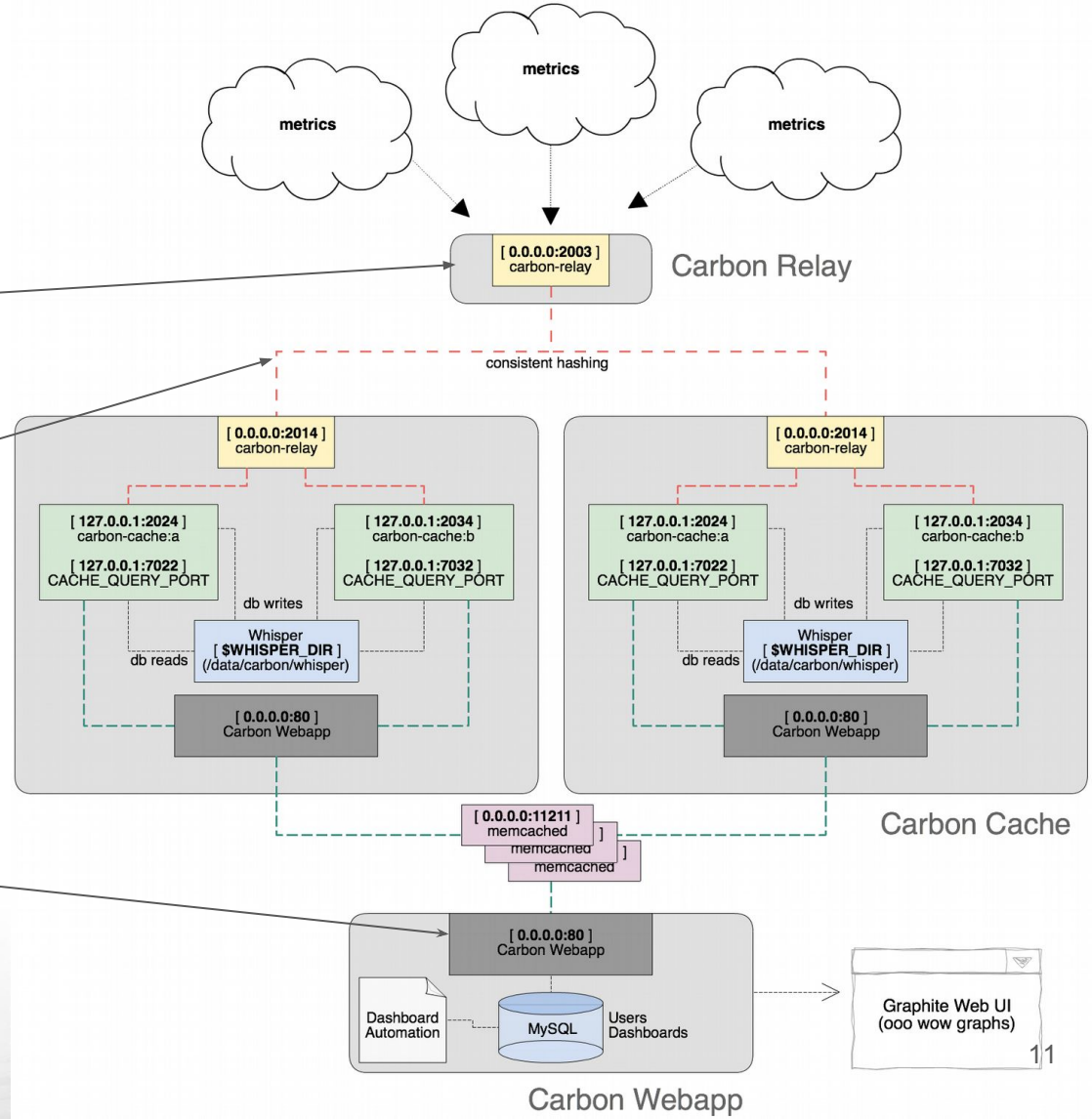
- Four Components
  - relay, cache, web & db
- Cluster Node:
  - 1 x Relay -> N x Caches
  - N x Caches -> 1 x DB
  - 1 x Web <- DB + Caches
- Inter-Node:
  - Arbitrary architecture of relays
    - (consistent hashing)
  - E.G. 1 incoming relay -> N x cluster node head relay
- DB: RRD-Like 1 file-per-metric (whisper)
  - Define retentions based on metric regexp
  - Directory structure is fast index
- Cache: holds points in memory on way to disk (carbon-cache)
  - Configurable write-IO profile, aggregates & groups metrics
- Web (graphite-web)
  - WSGI App, own GUI + REST API
    - Grafana talks REST
  - Queries to one webapp fan out and merge across cluster

Graphite Architecture

HAProxy can go here

Arbitrary layers of relays + replication factors here...

Grafana



Graphite Limitations

Multiple dimensions in the data bloat the metric space, for example:

- "condor.<exp>.<slot>.{ram, cpu, etc...}" may have 10 metrics in 15k slots in 3 experiments
  - 450k Metrics
- "condor.<exp>.<user>.<schedd>.{ram, cpu...}" may have 10 metrics, 500 users across 10 schedds in 3 experiments
  - 150k Metrics
- Now take the cartesian product of these, and you'll soon run out of inodes on your graphite cluster

Graphite Limitations

This sort of thing will drive us to look at multidimensional, tag-based TSDB

InfluxDB recently went "open-core" and made clustering enterprise only

(ノ °□°)ノ ⌒┻━┻

So... OpenTSDB? Do we *really* want HDFS + HBASE just for condor monitoring?

Prometheus? Ehh? Maybe?

At this point, living with graphite sounds reasonable for us

Ancient (policy) History

Mimicking "Queues" with HTCondor

STAR and PHENIX experiments: different ways to classify jobs

- +Job_Type = "xyz"
- Owner = "special"
- +HighMem = true
- Flocking: HomeExperiment vs Experiment

ATLAS

- AccountingGroups

Every site cares about monitoring a different aspect of their pool

HTCondor Monitoring

Collects condor jobs from startds

<experiment>.<slot-type>.<state>.<custom1…n>.<group>.<user>

Schedd + Collector + Negotiator stats

<experiment>.<daemon>.<stat-name>

User Priorities + Usage

<experiment>.<domain>.<group>.<user>.<usage|prio…>

- Tip: Keep wildcards “above” high-multiplicity points in the hierarchy

Custom Hierarchy

Heavily modified Fifemon to support custom hierarchies of classads

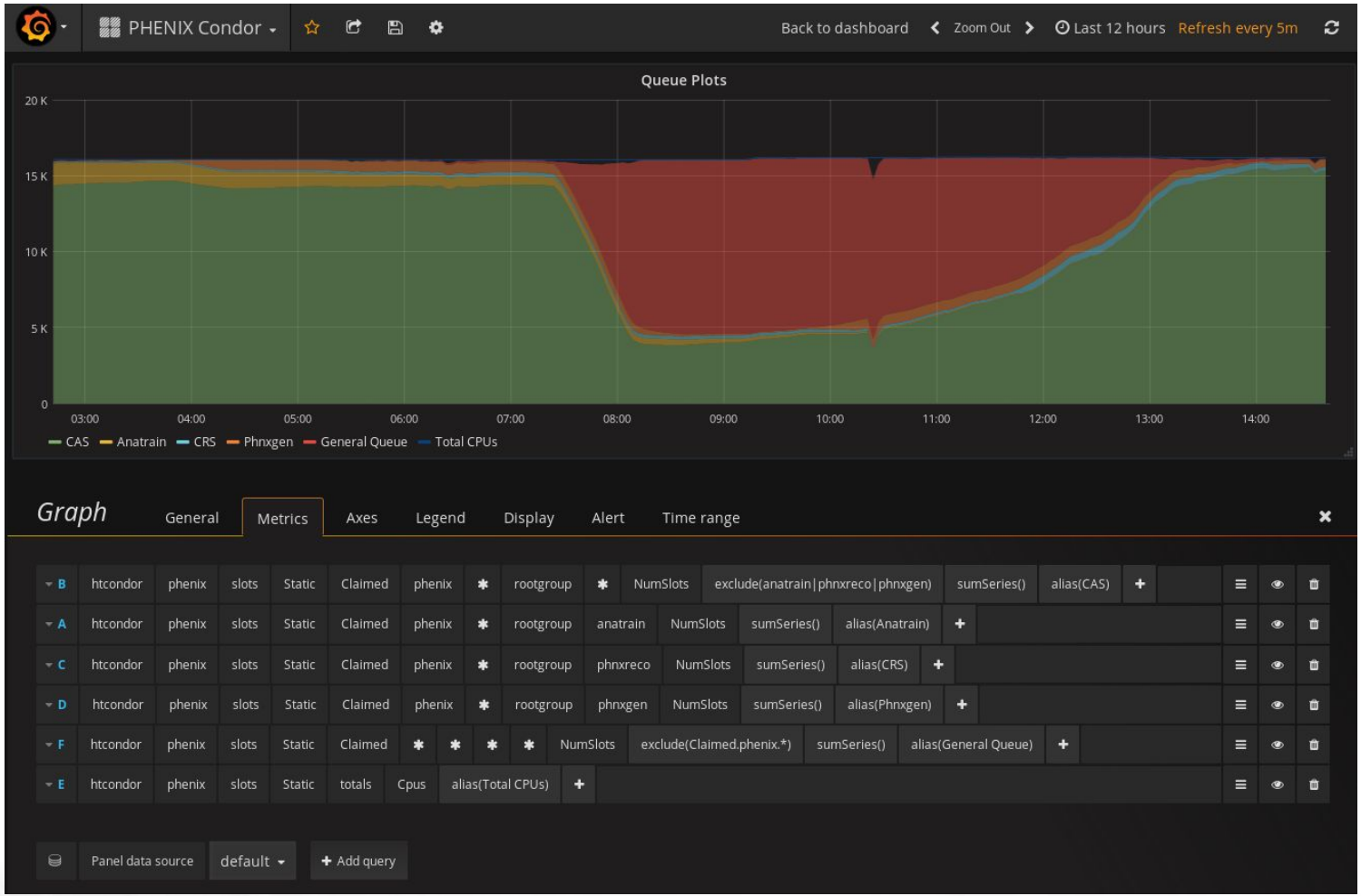
```
[star]
address = condor02.rcf.bnl.gov:9664
attrs = RealExperiment, Job_Type
groups = rhstar

[phenix]
address = condor01.rcf.bnl.gov:9662
attrs = RealExperiment, Job_Type
groups = rhphenix

[brahms]
address = condor02.rcf.bnl.gov:9661
attrs = CPU_Experiment, RealExperiment
groups = rhbrahms, dayabay, lbne, eic, astro
```

<experiment>.slots.<type>.<state>.<job-experiment>.<job-type>.<user>.num

phenix.slots.static.claimed.phenix.crs.*.cpus

phenix.slots.static.claimed.phenix.highmem.*.cpus

Status Dashboard



PHENIX Query

Direct CGroup Monitoring

Idea: mine the condor-generated cgroups for useful info to send to a TSDB

Why? Isn't all the info in the condor_schedd?

- Schedd updates only periodically, incomplete info compared to cgroup
- Google's cAdvisor? Go, embedded webserver, docker, ehhh
- My own solution? Small libcgroup-based script that sends data to graphite or statsd.

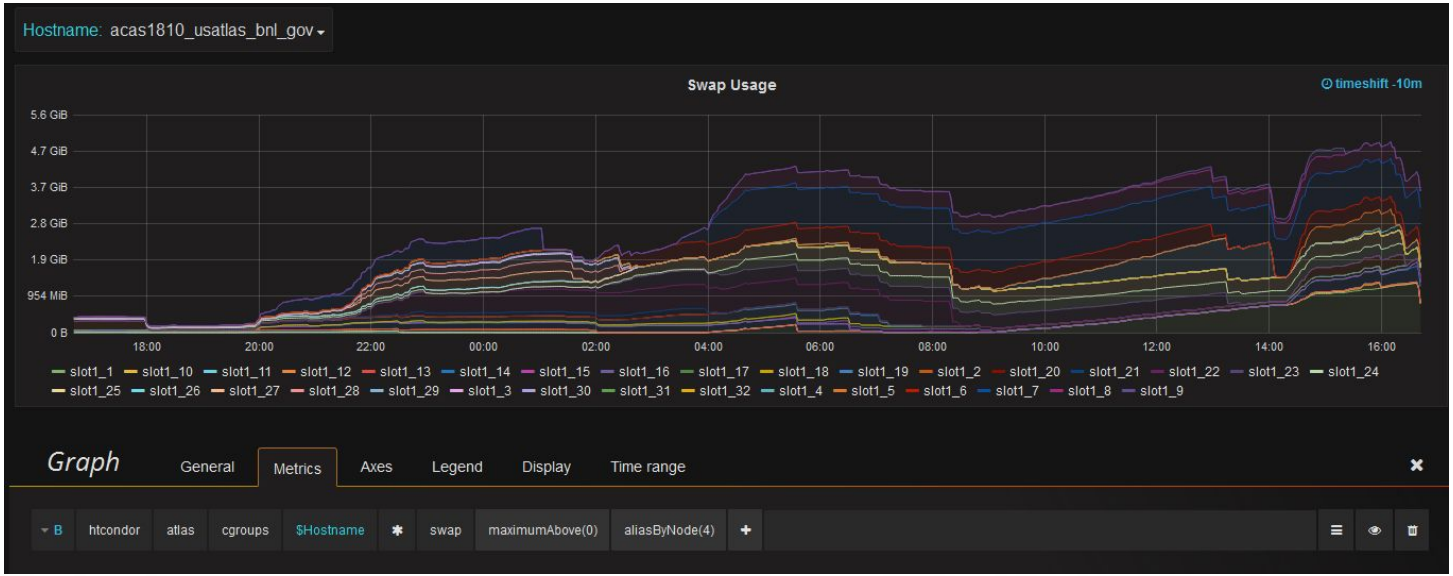
ATLAS CGroup Interface

CGroup Monitor

- Written in C, cmake RPM
  - Note: do not look into the depths of libcgroup!
- Reports to graphite via TCP or UDP
  - Also to StatsD
- Measurements per-job
  - Alert high memory / swap usage
  - Alert low CPU-efficiency
- Can be extended to measure IO per device, etc…
- Run under cron
  - puppet's fqdn_rand() for splay
- Integration: Collectd plugin? HTCondor?

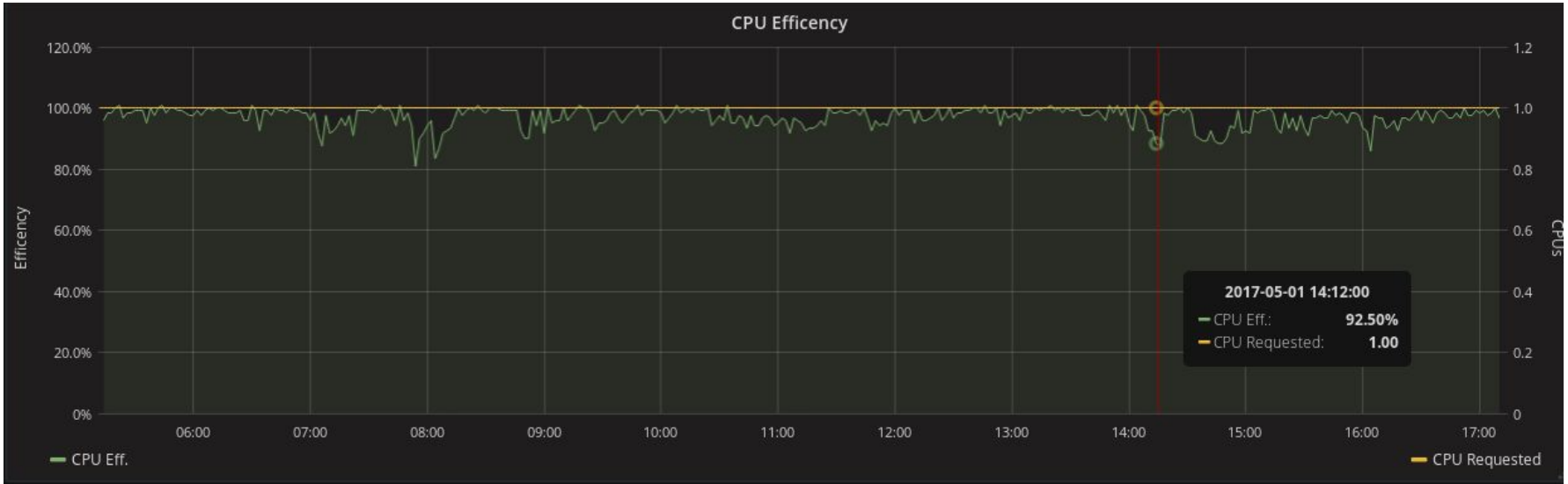
CGroup Monitor

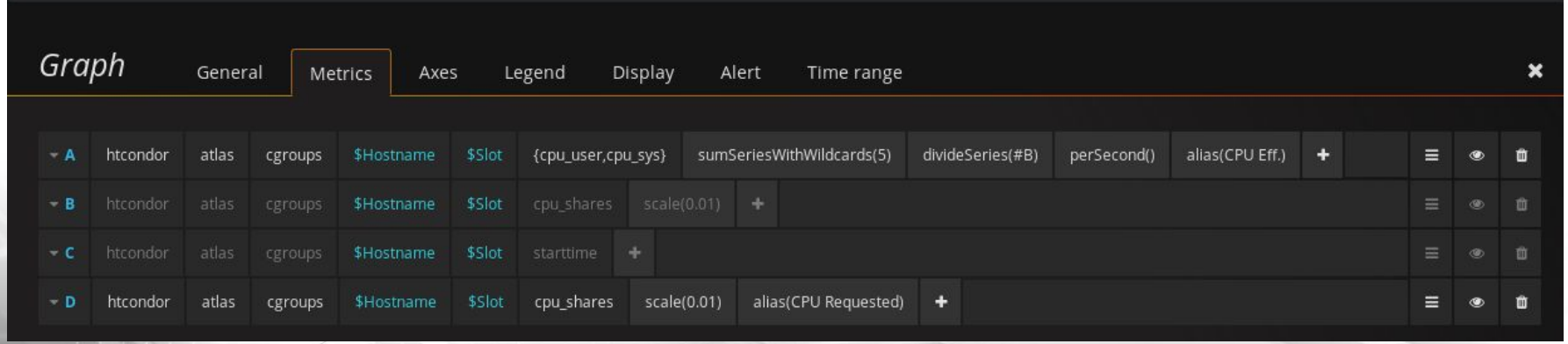
Who is swapping?



CGroup Monitor

How efficient is a job?





Conclusion

- Analytics is *not* accounting
  - We're interested in the former for now
- Graphite is a stable but effective choice of a TSDB
  - Good performance, great query interface via Grafana
- Fifemon is a step in the right direction
  - Integration with HTCondor would be great
  - Some extension done by me, happy to collaborate
- Custom cgroup monitoring proven useful for us
  - Where to put it?
- Need elasticsearch or similar for ingesting startd / schedd history logs
  - Area of current work for us

Thank You!

Questions? Comments?

We're hiring. Contact Eric Lancon or talk to myself or Tony Wong here this week.