RHIC relativistic heavy ion collider

RHIC & ATLAS Computing Facility
at Brookhaven National Laboratory

ATLAS EXPERIMENT

Monitoring with Clustered Graphite & Grafana

Metrics data analytics at the RACF for HTCondor & More

William Strecker-Kellogg

HEPiX—October 2016

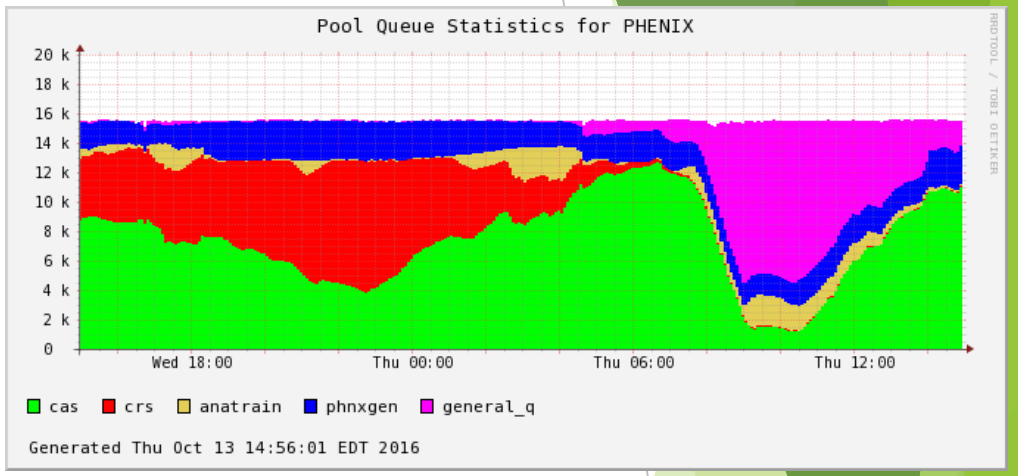
BROOKHAVEN NATIONAL LABORATORY

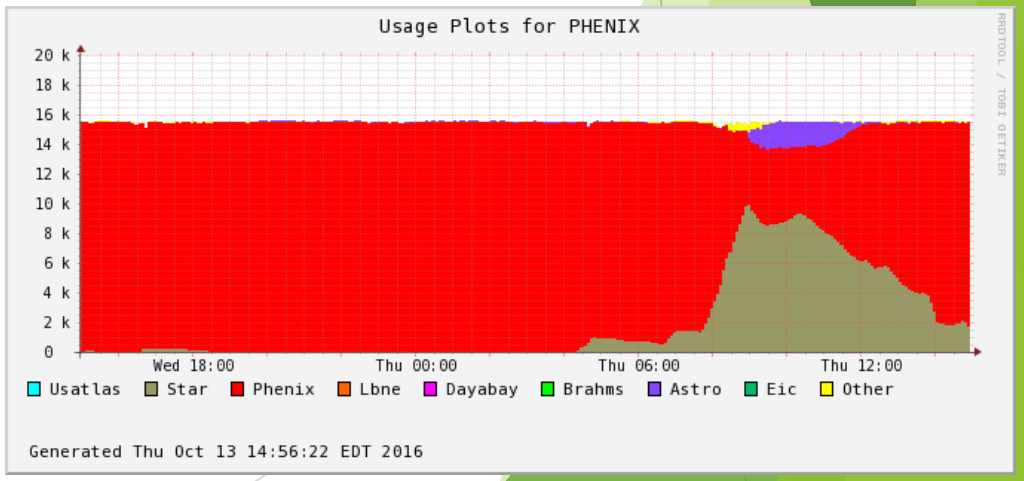
Overview

- Goals: Revamp ~~Batch~~ Monitoring
- Outline:
  1. Current monitoring status
  2. Evaluation of new tools for storage, presentation, & collection of metrics
     - Time-Series Database selection, Frontend selection, & metrics-gatherer selection
  3. HTCondor-specific metrics collection
     - 2 Tools and examples
  4. Conclusions

Current Condor Monitoring

- RRDTool + Scripts
  - Custom python script gathering
  - What to gather—hardcoded in scripts!
  - Remap q1…q64 → meaningful names @ plot-time (messy)
- Ganglia
  - Stable, not going anywhere, OK, …





Current Condor Monitoring

```
#!/usr/bin/python

# ****************************************************************
# For graphing each queue of an experiment as defined in the dictic
#

import logging
import common

qmap = {
    'star': (
        ("cas", {"job_type": "cas", "RealExperiment": "star"}),
        ("crs", {"job_type": "crs", "RealExperiment": "star"}),
        ("osg", {"job_type": "osg", "RealExperiment": "star"}),
        ("long", {"job_type": "long", "RealExperiment": "star"}),
        ("high", {"job_type": "high", "RealExperiment": "star"}),
        ("general_q", {"RealExperiment": "!star"}),
        ("num_cpu", {"turn_off": "false"}),
    ),
    'atlas': (
        ("prod", {"racf_group": "production"}),
        ("amc", {"racf_group": "amc"}),
        ("long", {"racf_group": "long"}),
        ("short", {"racf_group": "short"}),
```

- Our monitoring script
  - Uses a custom parser/query language for classads
  - Configuration hardcoded
  - Just look at how great it is!...
    - Plotting is worse
- But… how we classify jobs is unique per-experiment

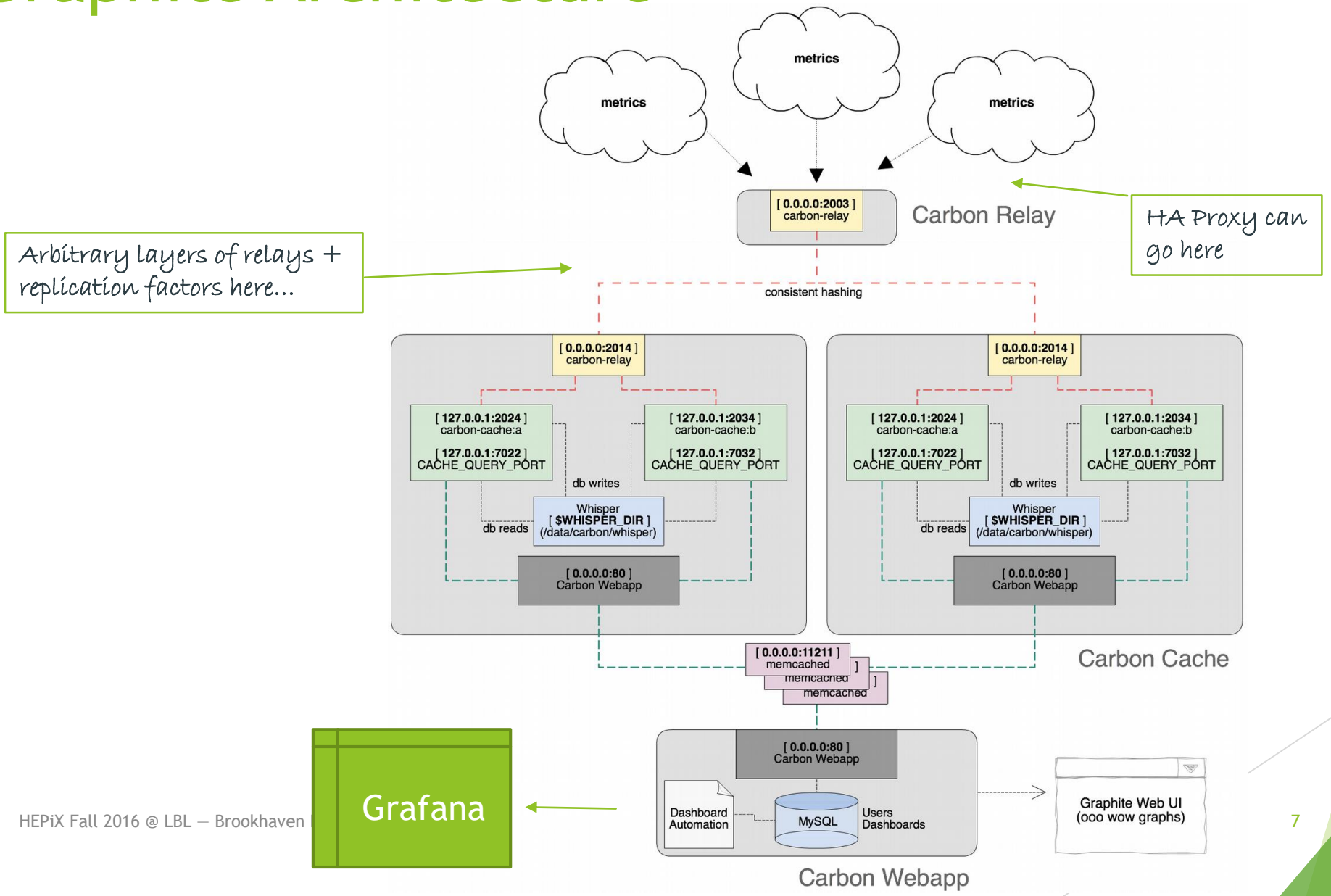
Evaluating Time-Series DBs

- InfluxDB
  - Went “open core”: clustering only for paid product… (ﾉ°□°)ﾉ︵┻━┻
- OpenTSDB
  - Complex setup…requires Hadoop/HDFS/Hbase
- Graphite
  - Simple design, easy deployment
  - Performant (50kHz+)
  - Best query API
- Statsd
  - Not a TSDB, but a useful aggregator / cache / frontend to several TSDBs

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
  - Any architecture of relays (consistent hashing)
- Upstream
  - Direct Clients (Collectd, Ganglia, etc…)
  - Statsd
- DB: RRD-like, 1 file per metric (whisper)
  - Directory structure is fast index
- Cache: holds points in memory & throttles disk IO (carbon-cache)
  - Configurable IO load, responds to web queries
- Web: (graphite-web)
  - WSGI App, GUI and REST API (used by Grafana)
  - Merges results from across cluster

Graphite Architecture



Frontend: Grafana

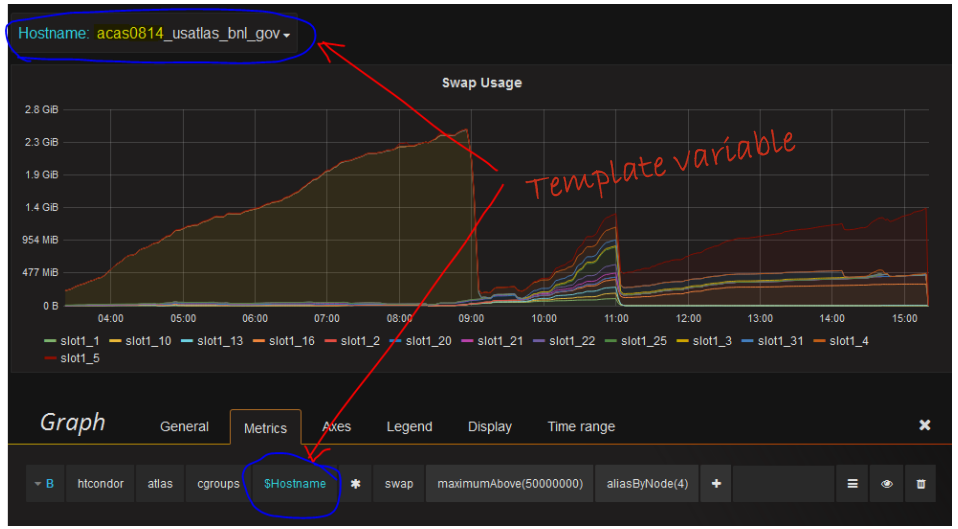
Context

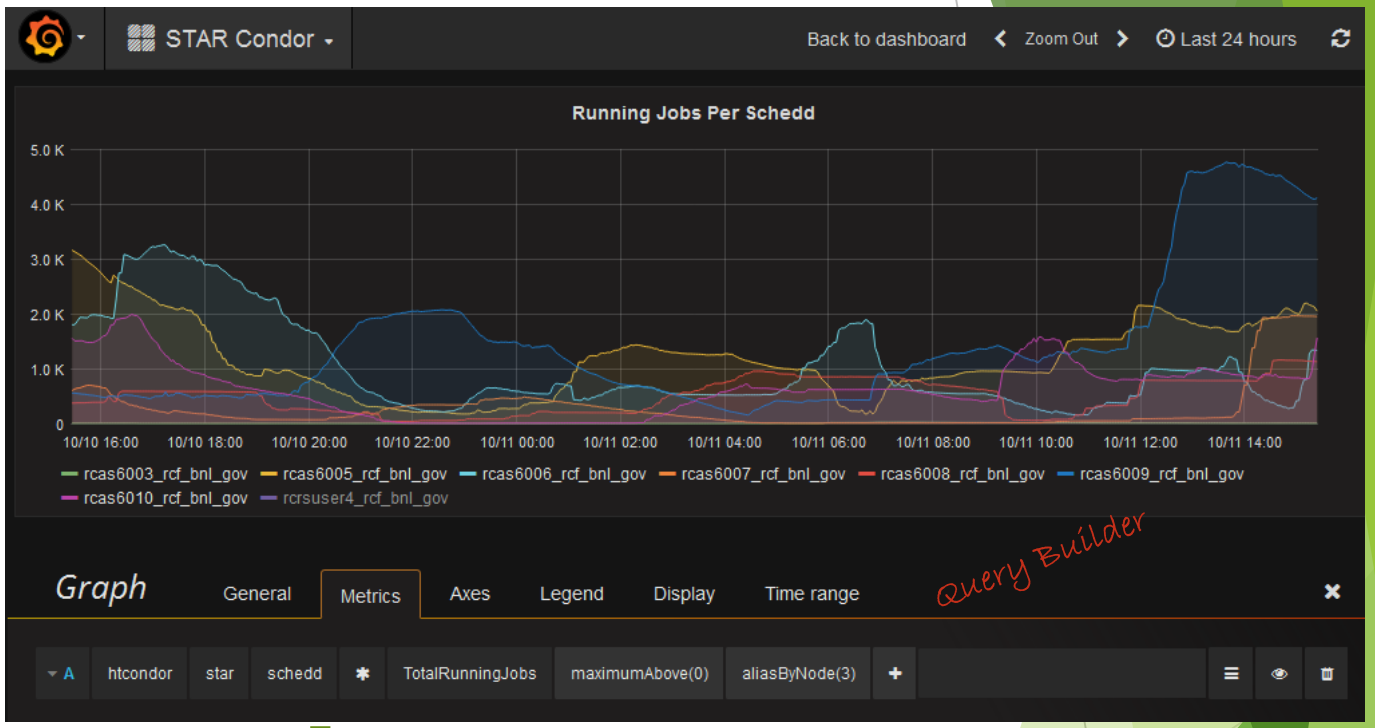
- Kibana vs Grafana ←→ Elasticsearch vs Graphite
  - Logs vs Metrics
- Grafana grew out of Kibana
- Main goal is dashboard creation & sharing
  - Share socially
  - Define Alerts
  - Templating on variables and sources

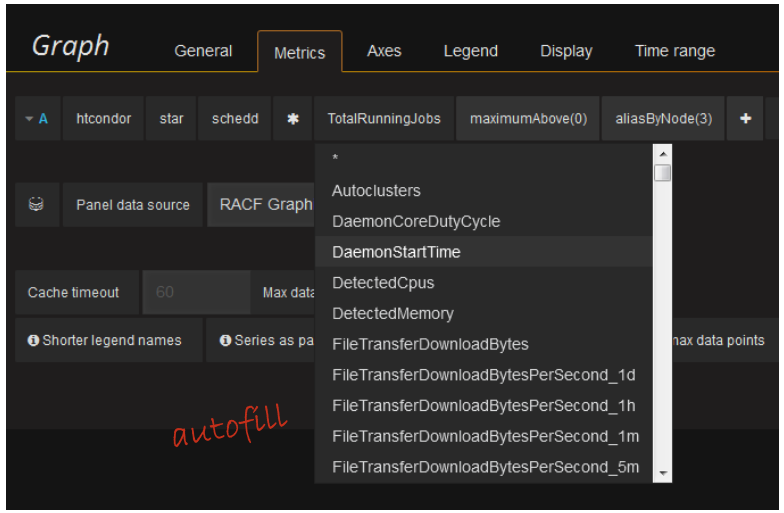
Features

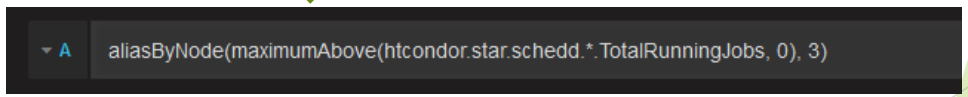
- Users & Orginizations
  - User→Org. many-to-many
  - But…no multi-org dashboards
- Data Sources
  - proxy or passthrough, but per-org
  - Influx, Promethius, Graphite, Elasticsearch, OpenTSDB & more…
- Flexible Auth
- Dashboards
  - Permissions can be RO, Copy, RW

Query Interface









(edit query string)

Grafana Dashboards

Evaluating Data Collection

- Ganglia has graphite component
  - Don't use TCP mode!
  - Want application-aware monitoring + customizable framework
- Collectd has many, many plugins
  - Somewhat complex to build packages
  - Widely used, large community, well supported
- Diamond has many plugins
  - Extensible with custom plugins
  - Handlers for sending data to dozens of places
  - Fairly widely used, etc…

Evaluating Monitoring Clients (HTCondor)

- HEPiX Batch Monitoring working group
  - Gathering data from the community
  - No community standard, current solutions are generally patched together
- Central vs Distributed Collection
  - Write plugins for ganglia/collectd?
  - In HTCondor, many things in the collector
    - Exception: schedd, but O(10) schedd/exp; therefore easy to poll
  - What do we care about
    - How many jobs of each type/user over time
      - Running and Idle (where / how many)
    - Performance self-reporting

HTCondor Monitoring

- Fifemon Project
  - Developed at Fermilab
  - Monitors HTCondor by polling daemon & slot usage-statistics
- Heavily modified by me to allow custom hierarchies of ClassAds
  - Only wanted some data
  - Certain classads we need to monitor
- Reports to Graphite & InfluxDB

```
[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
```

HTCondor Monitoring

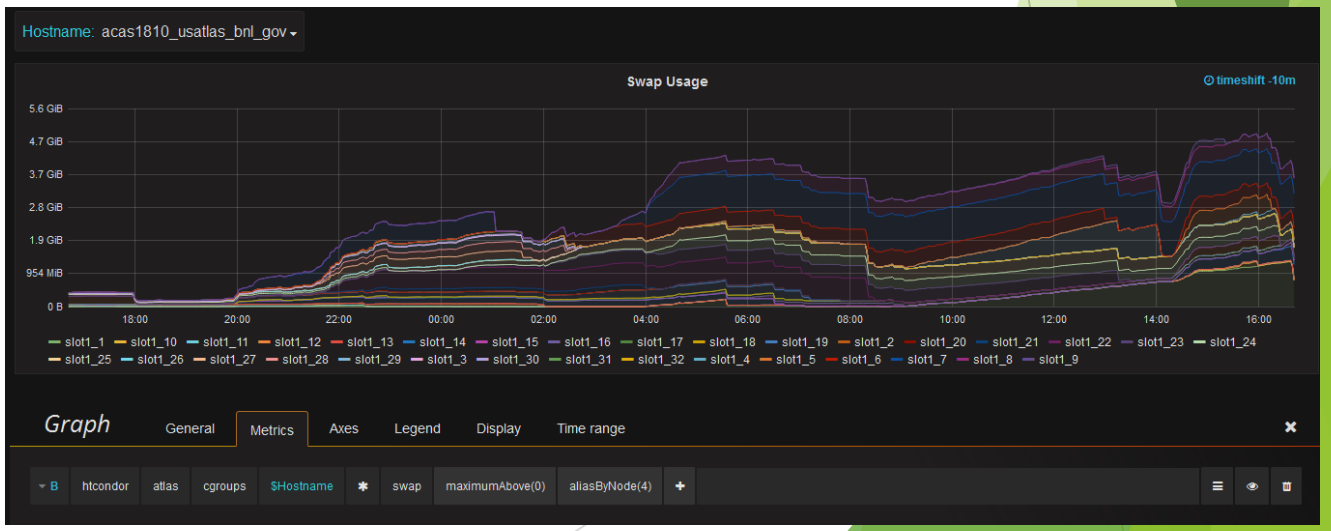
- Collects condor jobs from startds
  - <experiment>.<slot-type>.<state>.<custom1…n>.<group>.<user>
- Schedd + Collector + Negotiator stats
  - <experiment>.<daemon>.<stat-name>
- User Priorities + Usage
  - <experiment>.<domain>.<group>.<user>.<usage|prio…>

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

HTCondor Resource Usage Monitoring (CGroups)

- Condor keeps track of some resourse usage
  - CPU time & Memory
- Can use CGroup-based process tracking
  - CGroups “know” much more
  - Swap, efficiency, IO, etc…
  - Per job per node

- I wrote a program to report HTCondor Cgroup data to Graphite
  - Writes to graphite or statsd



HTCondor Resource Usage Monitoring (CGroups)

Why

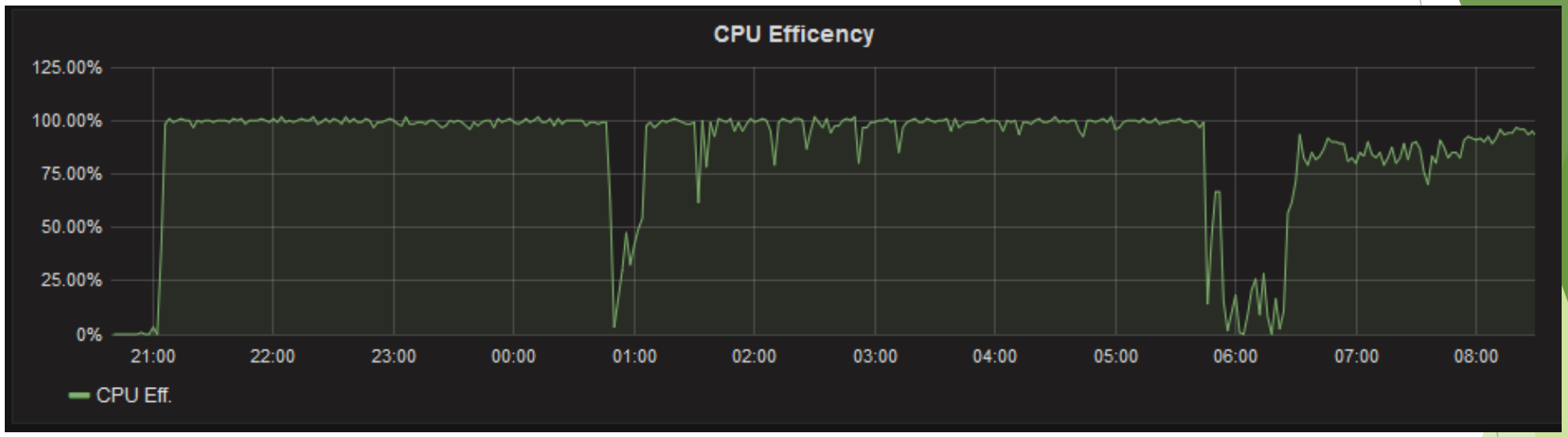
- Condor's view into User/Sys_cpu, RSS, etc… is incomplete
  - Delayed updating
  - Incomplete picture
- CAdvisor
  - Google project to monitor CGroups
  - Powerful…and overkill
    - Docker container, embedded webserver, etc…

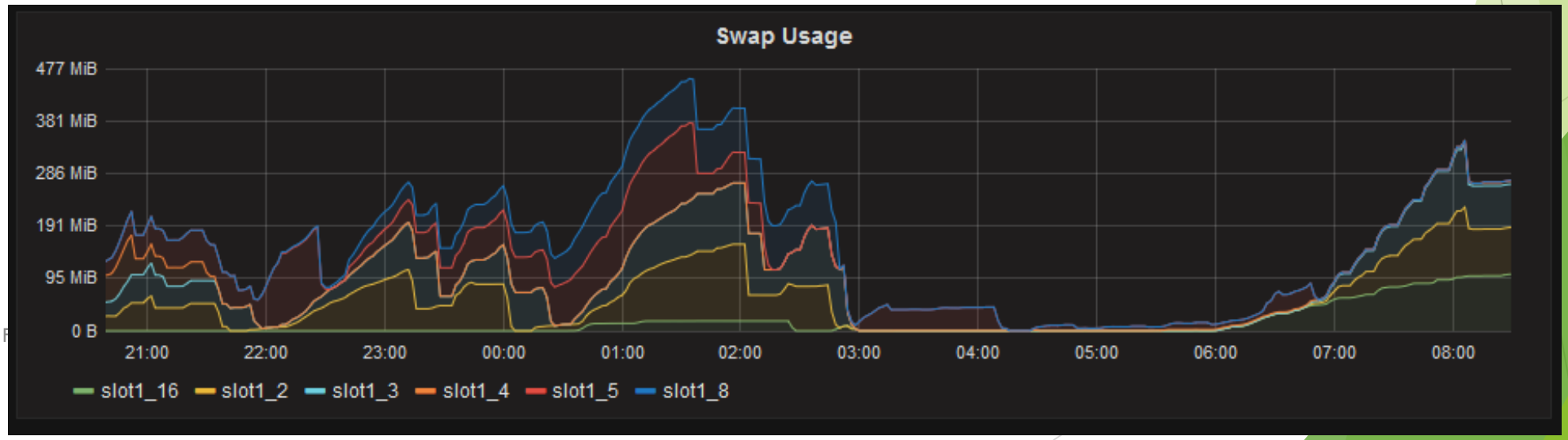
How

- 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

HTCondor Resource Usage Monitoring

(Examples)





Conclusions (opinions)

- Better to use well-supported community solutions
- Graphite is a stable but effective choice of a TSDB
  - Good performance, most powerful query interface
- Grafana is the most flexible dashboard
- HTCondor monitoring is cludgey but Fifemon is a step in the right direction
  - Is condor_gangliad going to become more general purpose?
  - More work is needed on batch monitoring—HEPiX Group
- For collection, builtin application-specific monitoring is nice…
- … ability to add custom monitors is nicer!
  - ongoing evaluation / deployment of Diamond & Collectd

Questions? Comments?

Thank You For Your Attention!

William Strecker-Kellogg <willsk@bnl.gov>