Monitoring And Accounting For Cloud Resources

Frank Berghaus
work by
Rolf Seuster (UVic)
Fernando Galindo (TRIUMF)

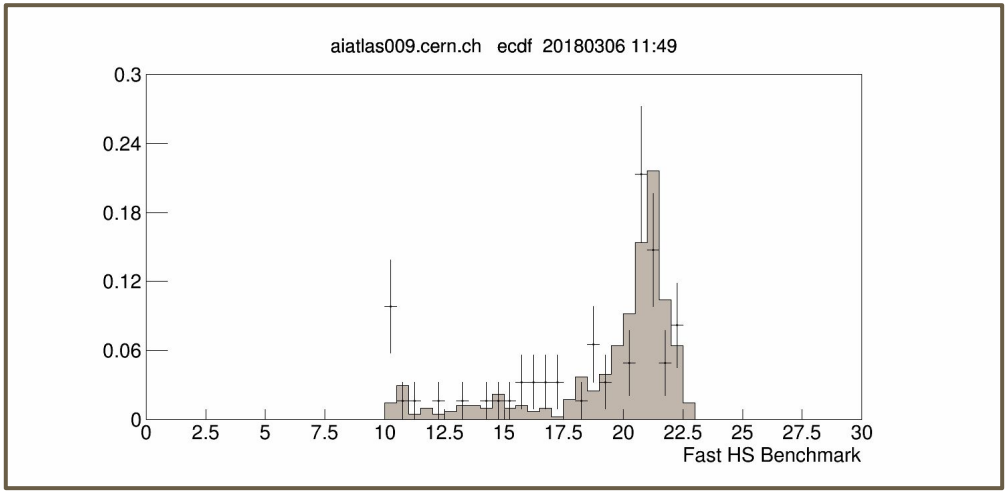
Rolf Seuster

Job Monitoring and Accounting

Run fastBmk (HEPiX) from cvmfs

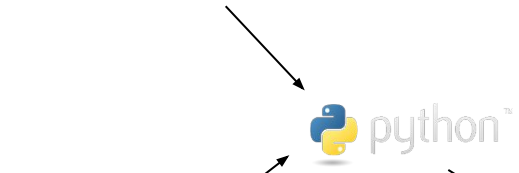
Collect fastBmk, user and system time

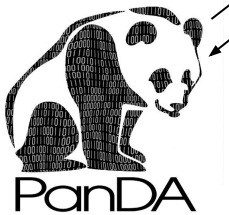
Publish daily, weekly, and monthly values on VM shutdown



CloudScheduler:
-name of VM
-cloud

HTCondor:
-name of VM
-GlobalJobId
-'VMType'





Dynafed Monitoring

Fernando Galindo

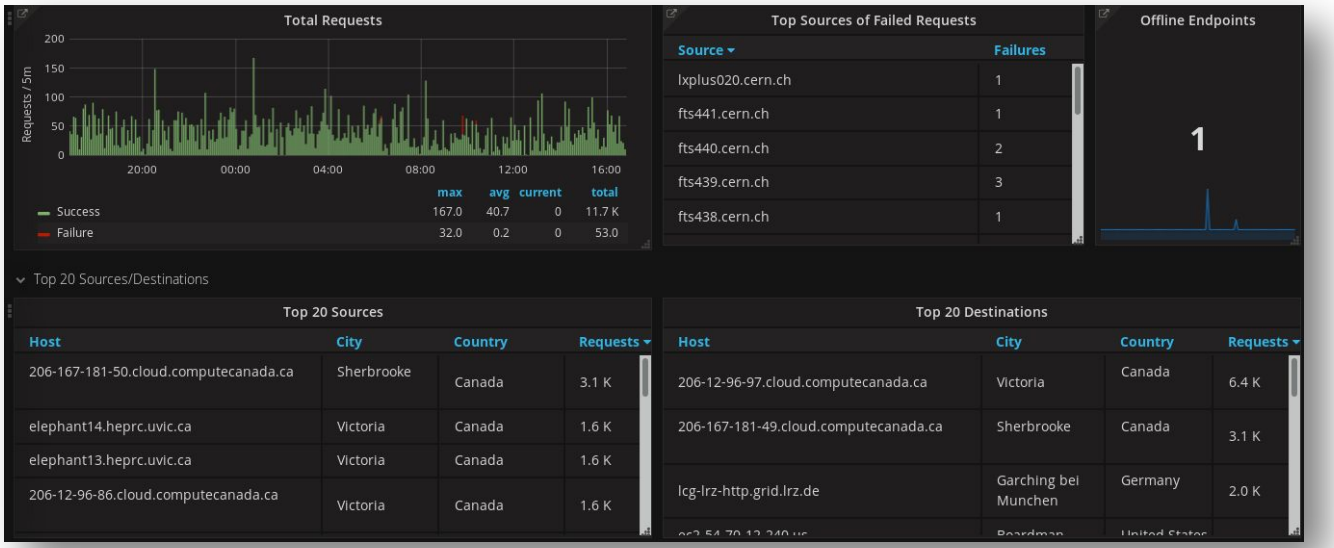
Apache
- *Dynafed*
- *Produces logs*

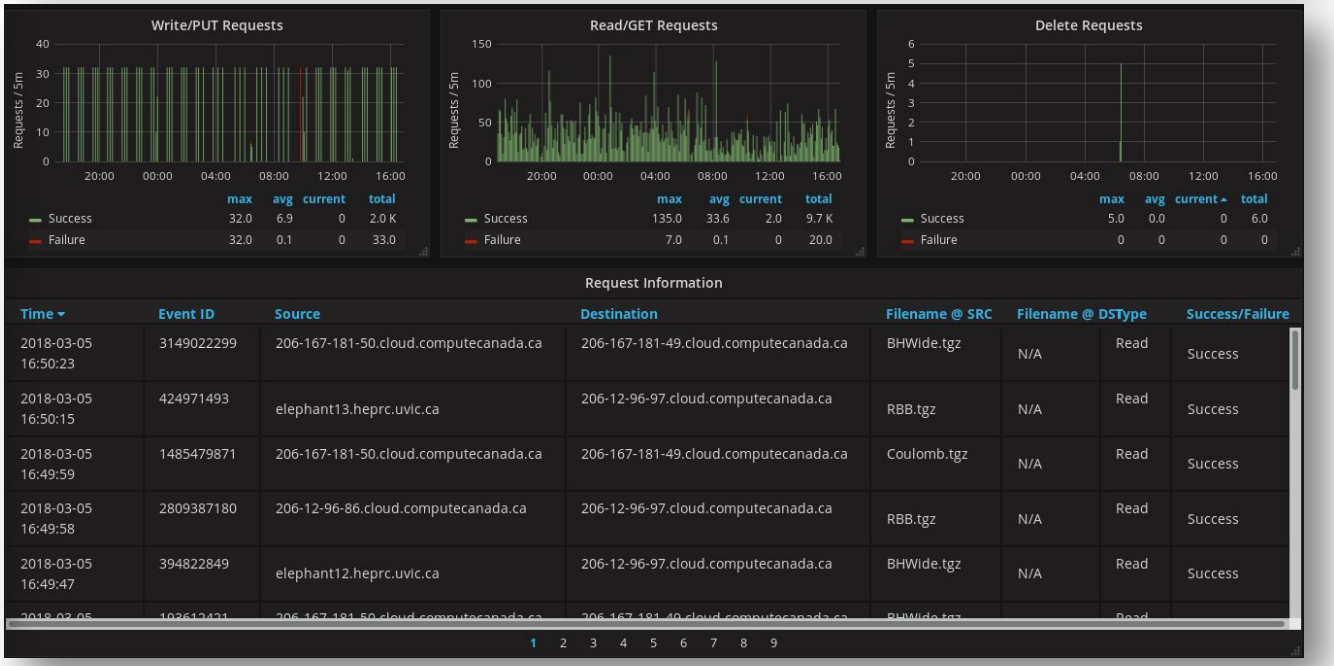
Filebeat
- Monitors log files
- Exports events as JSON
- Outputs to Logstash

Logstash
- Aggregates events
- Parses and enriches
- Outputs JSON docs

ES
- Stores
- Indexes
- Replicates

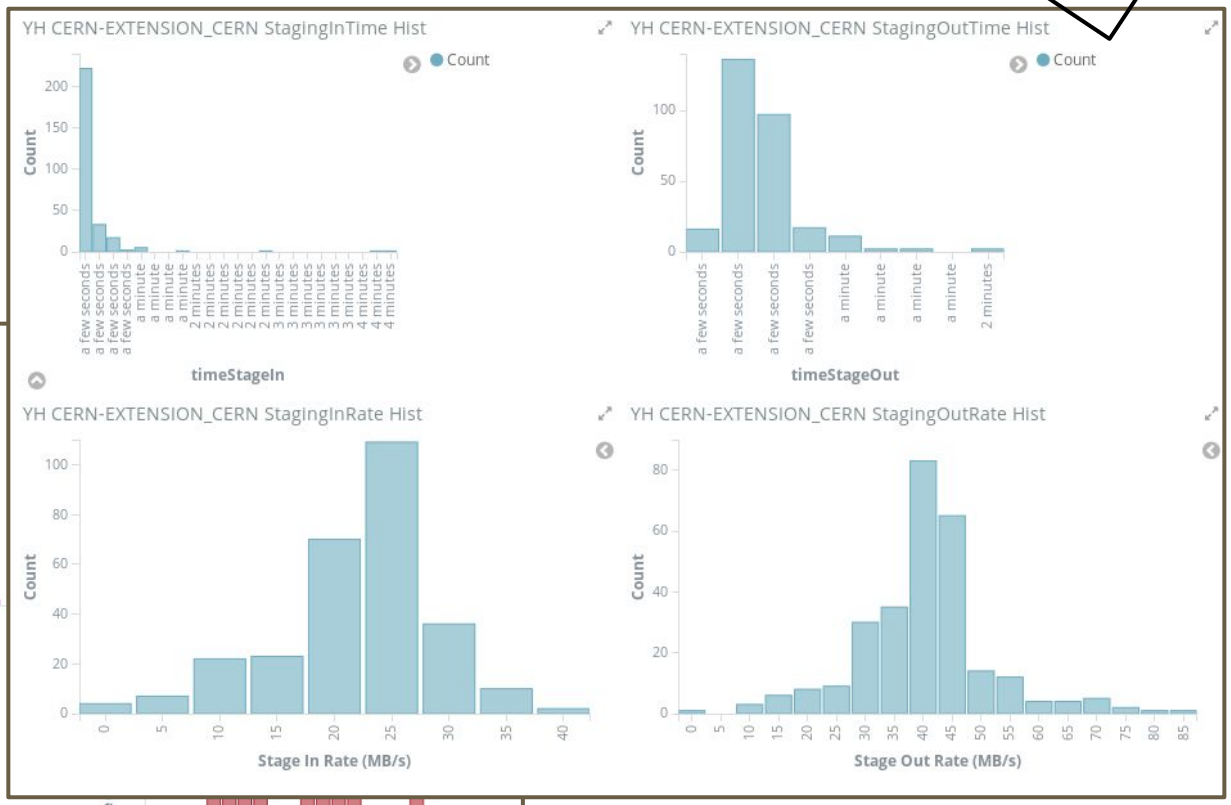
Grafana
- Visualizes

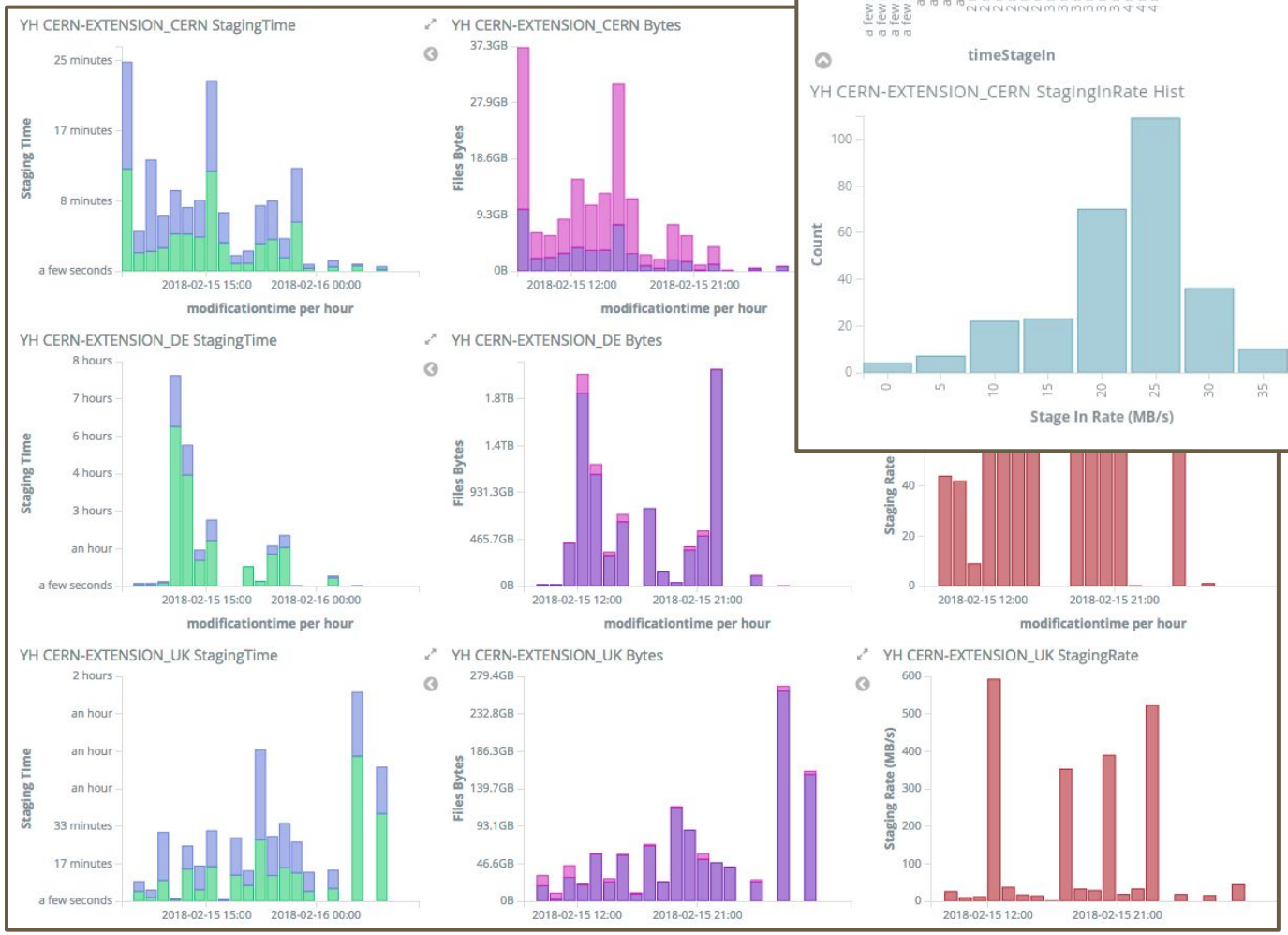




Job Staging
(with Dynafed)

Yun-Ha Shin





- Staging times and data rates
- Jobs run on cloud resource
- Data Access:
  - Local grid SE
  - Local grid SE via Dynafed
  - Object Store via Dynafed

Summary

1. Cloud Monitoring and accounting:
   a. At CERN: https://es-cloudmon.cern.ch
   b. At UVic: https://elk.heprc.uvic.ca:15601
   c. Scripts: https://gitlab.cern.ch/seuster/ES-cloud-jobmonitoring
2. Dynafed Monitoring:
   a. At TRIUMF: https://atlas-fed-metrics.triumf.ca
3. Qualification Tasks Starting:
   a. Benjamin Rottler (Freiburg, PhD) - HTTP/Dynafed Benchmarking
   b. Benoit Roland (Freiburg, PostDoc) - Cloud Monitoring & Accounting (w. ROCED)

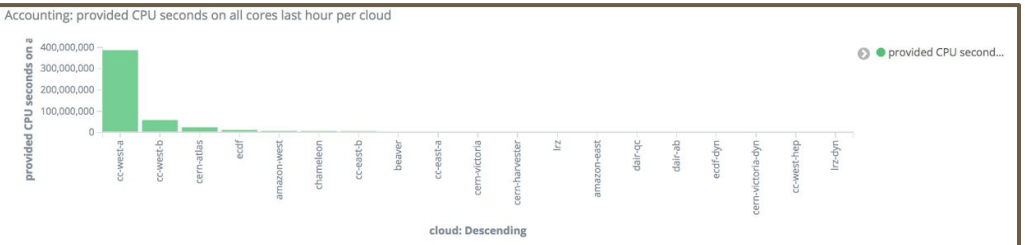
Backup & Details

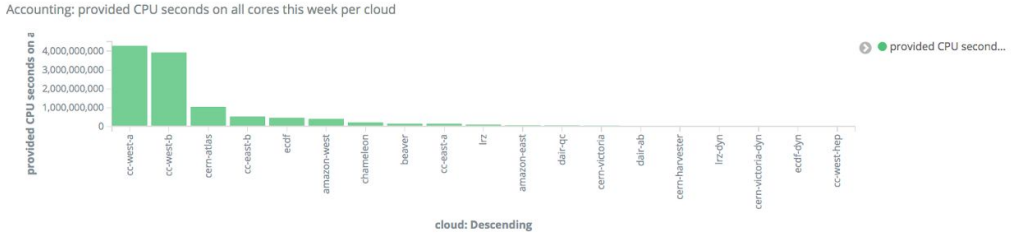
Job Accounting

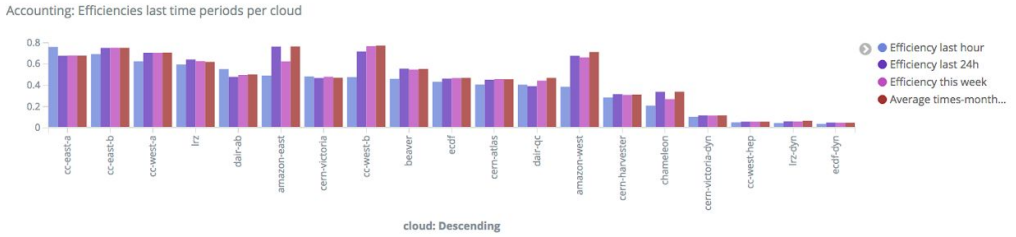
- Running many clouds for other groups
  - Split up resources by: new queue or accounting on our side
- Cross check commercial providers
- Compare clouds and focus on most performant ones
- Benchmark:
  - fastBmk (HEPiX benchmarking working group) python scripts from cvmfs
  - Run at boot of every VM, gives us rough estimate of HepSPEC06

Accounting Overview (2018 so far)

basic Cloudinformation: current and in 3 different time windows







Accounting Table CPU Hours (HS06) monthly

cloud: Descending	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
amazon-east	25.059	206.092	65.438	0	0	0	0	0	0	0	0	0
dair-ab	103.08	267.932	56.984	0	0	0	0	0	0	0	0	0
dair-qc	163.496	396.813	83.127	0	0	0	0	0	0	0	0	0
lrz	78.12	741.539	137.244	0	0	0	0	0	0	0	0	0
cc-east-a	645.404	845.596	14.625	0	0	0	0	0	0	0	0	0
amazon-west	346.629	1.497.433	218.788	0	0	0	0	0	0	0	0	0

efficiency + accounting table

to allow user to apply filters graphically by clicking

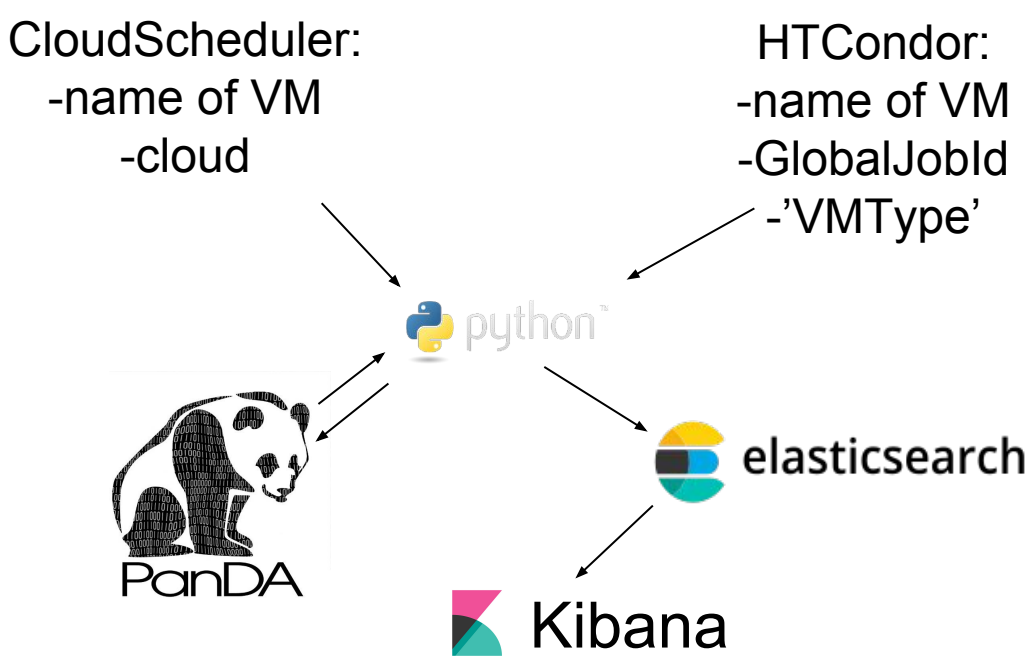
Job Accounting - Details

- Kibana: great interactive plots awful for processing
  - Process most numbers already on VMs and report only few stats
  - Store various numbers in pickle files on VM
    - to calculate CPU time provided this month, this week, this day
    - e.g. to get weekly numbers subtract numbers on Sunday at midnight from ‘now’
- Report system and user times → efficiencies
- Report monthly numbers in kibana table
- Ongoing: validate statistics in elasticsearch vs our old accounting system - does ATLAS publish own numbers ?

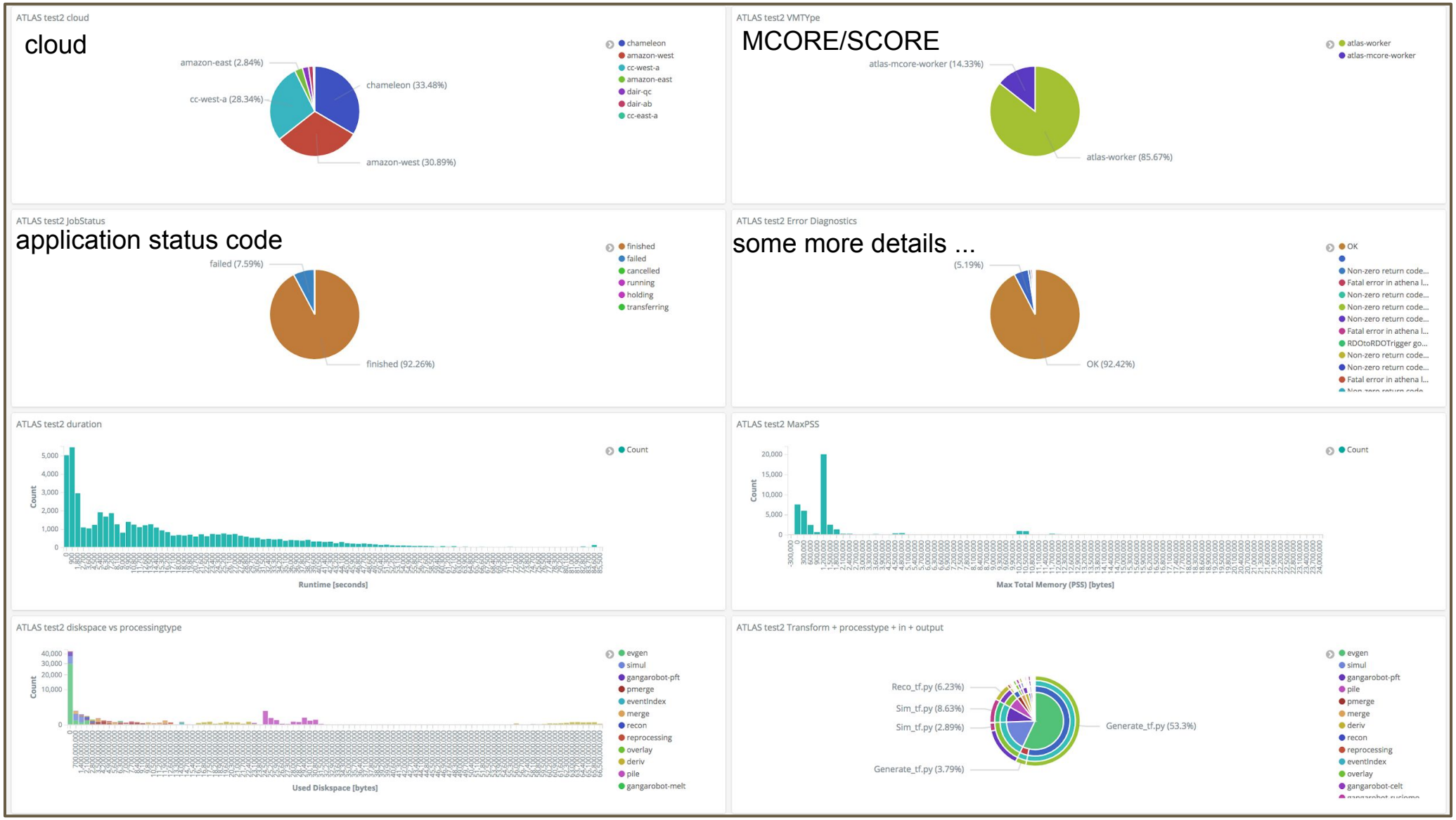
Job Monitoring

Work in Progress

- To aid debugging, get application status of workload from bigpanda
- Combine several sources of information to make useful for us:
  - CloudScheduler + HTCondor write to ascii file
  - python + bigpanda talk via REST + json
  - ES python bindings to upload to ES
  - Kibana to visualize



Cloud Job Monitoring (2018 so far)



Links

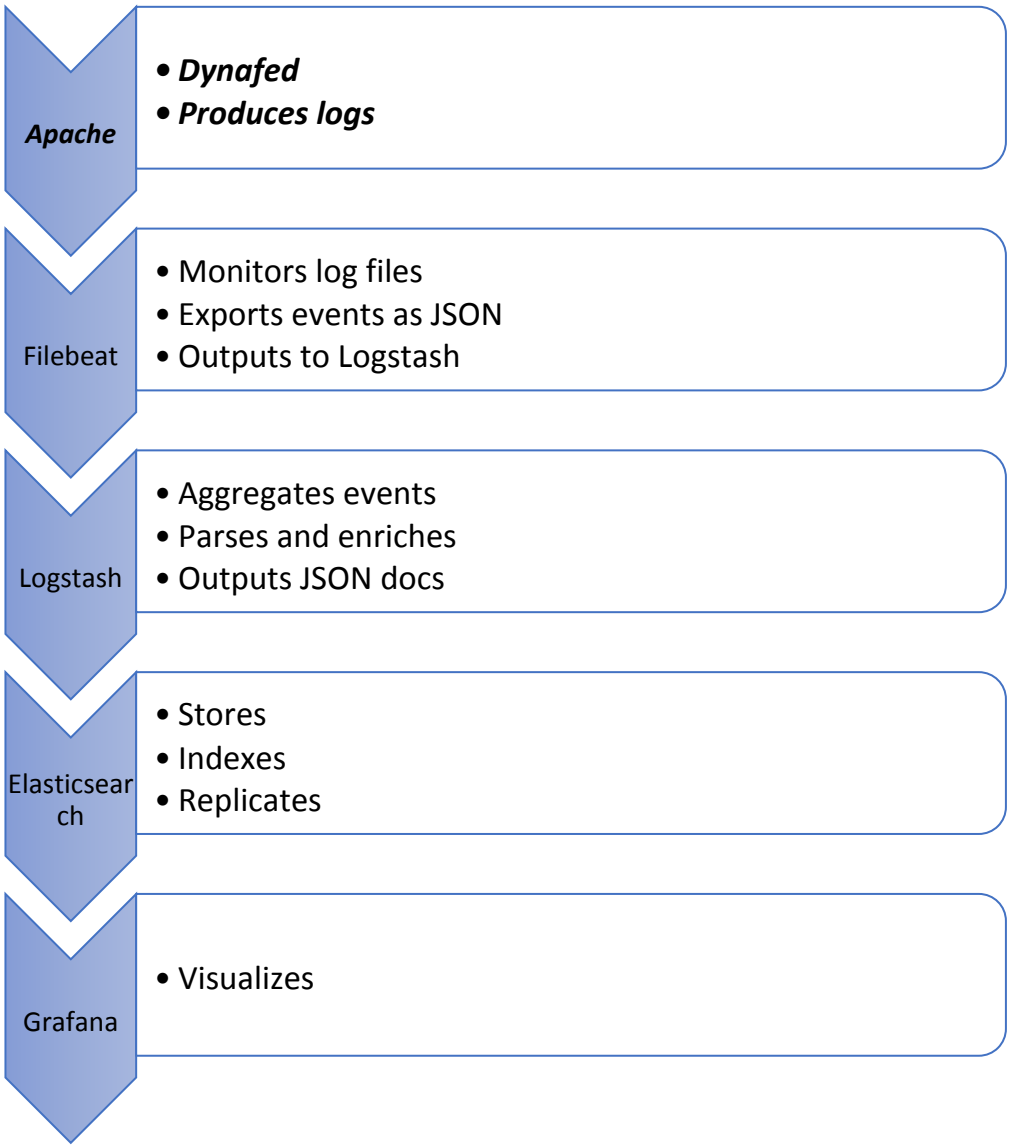
- ES at CERN (behind SSO):
https://es-cloudmon.cern.ch
- ES at UVic (mirror, testing and newer items):
https://elk.heprc.uvic.ca:15601
- CERN gitlab for monitoring code:
https://gitlab.cern.ch/seuster/ES-cloud-jobmonitoring
- Brief description of what's been done in more details (in progress):
https://wiki.heprc.uvic.ca/twiki/bin/view/HEPrc/AtlasJobMonitoring

Dynafed

Log Processing and Accounting

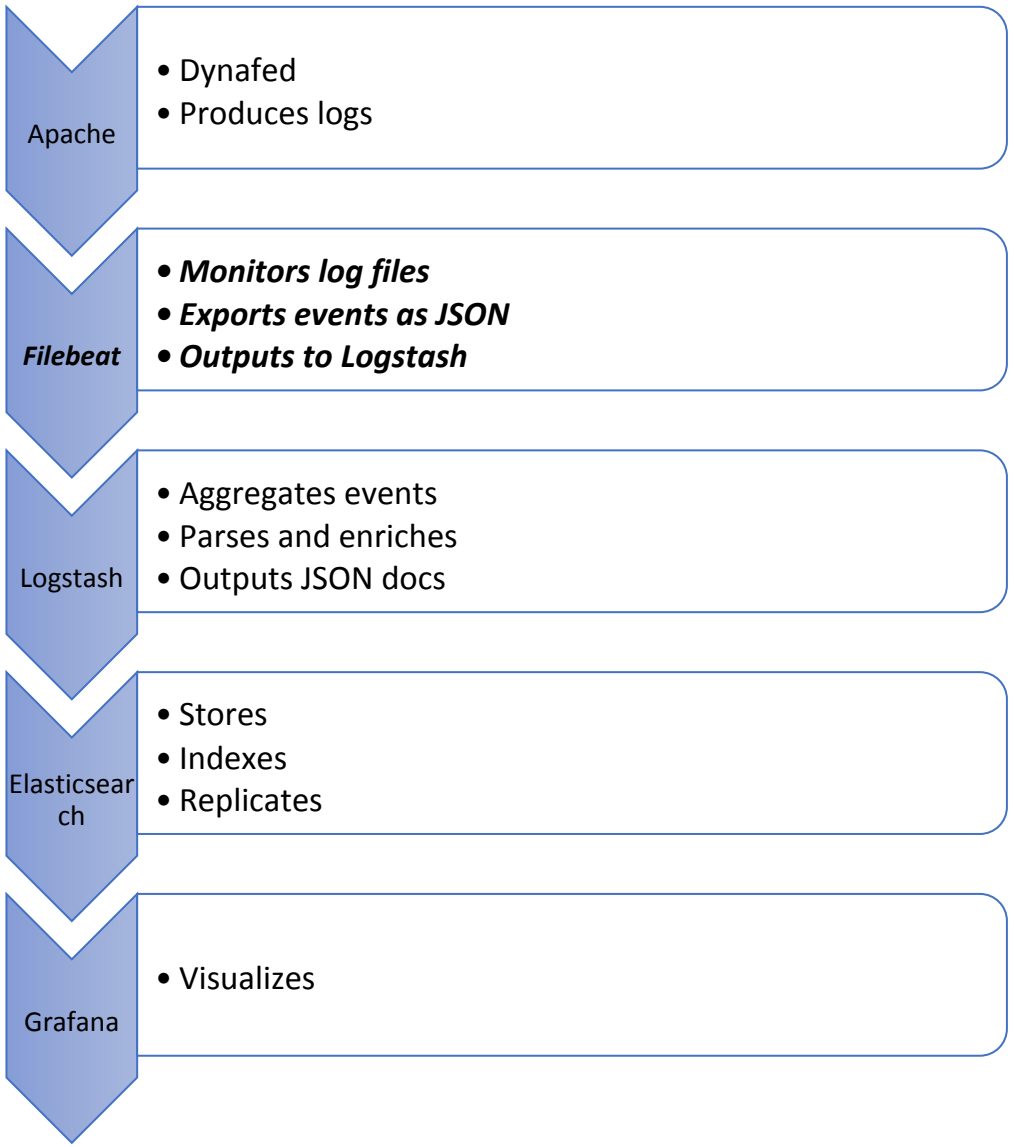
ELK – Stack Process - Apache

- Clients send GET, PUT or DELETE requests.
- Dynafed selects a storage endpoint and provides a 302 "redirect" link.
- Dynafed in sot aware of the transfer itself.
- This information is logged into two different files: access_log and error_log.
- The default format does not give enough information therefore we use custom formats and log files.



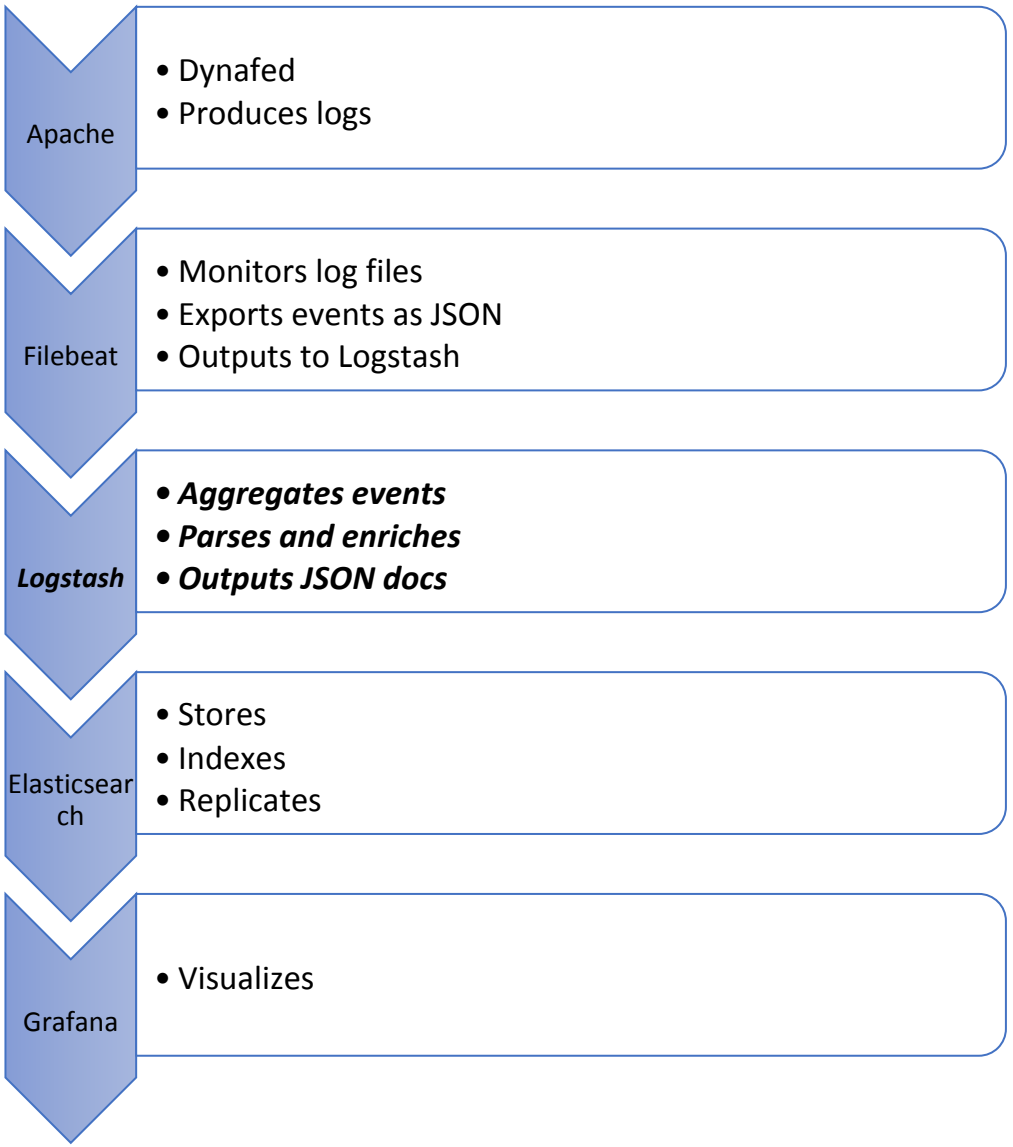
ELK – Stack Process - Filebeat

- Monitors Apache's log files.
- Queues new log lines.
- Embeds each line into a single JSON object field called "message".
- Sends objects to a Logstash for further processing.



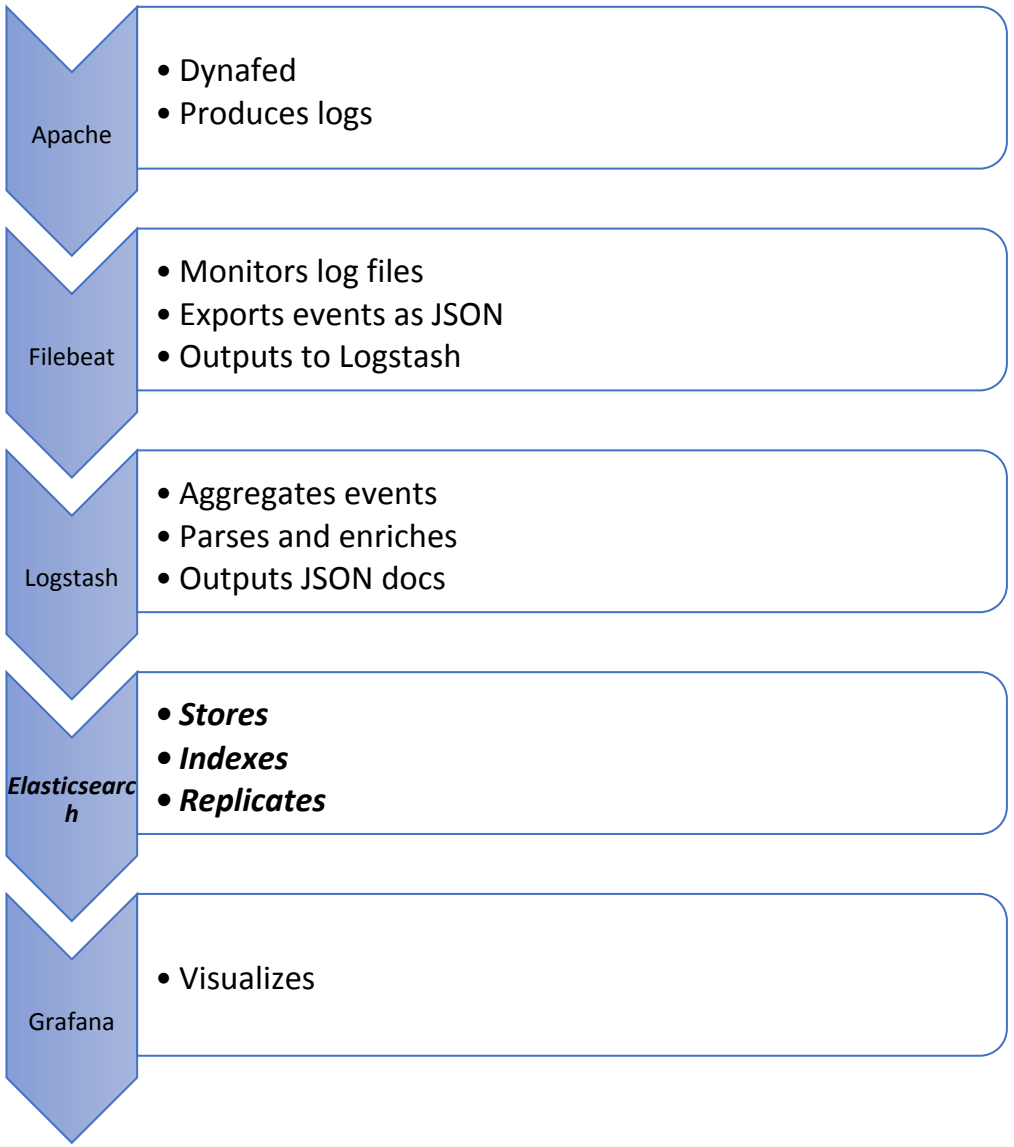
ELK – Stack Process - Logstash

- Parses each "message" and extracts all the data into key -> value JSON fields.
- Aggregates corresponding access and error log lines of each unique request.
- Extracts and sets the earliest timestamp.
- Determines type of request: read (GET), write (PUT), delete (DELETE) event.
- Determines success/failure on obtaining a redirect link.
- Extracts the client, requested file and source and destination storage endpoints.
- Resolves DNS and obtains GeoIP info.
- Creates two JSON Objects sent to Elasticsearch databse.



ELK – Stack Process - Elasticsearch

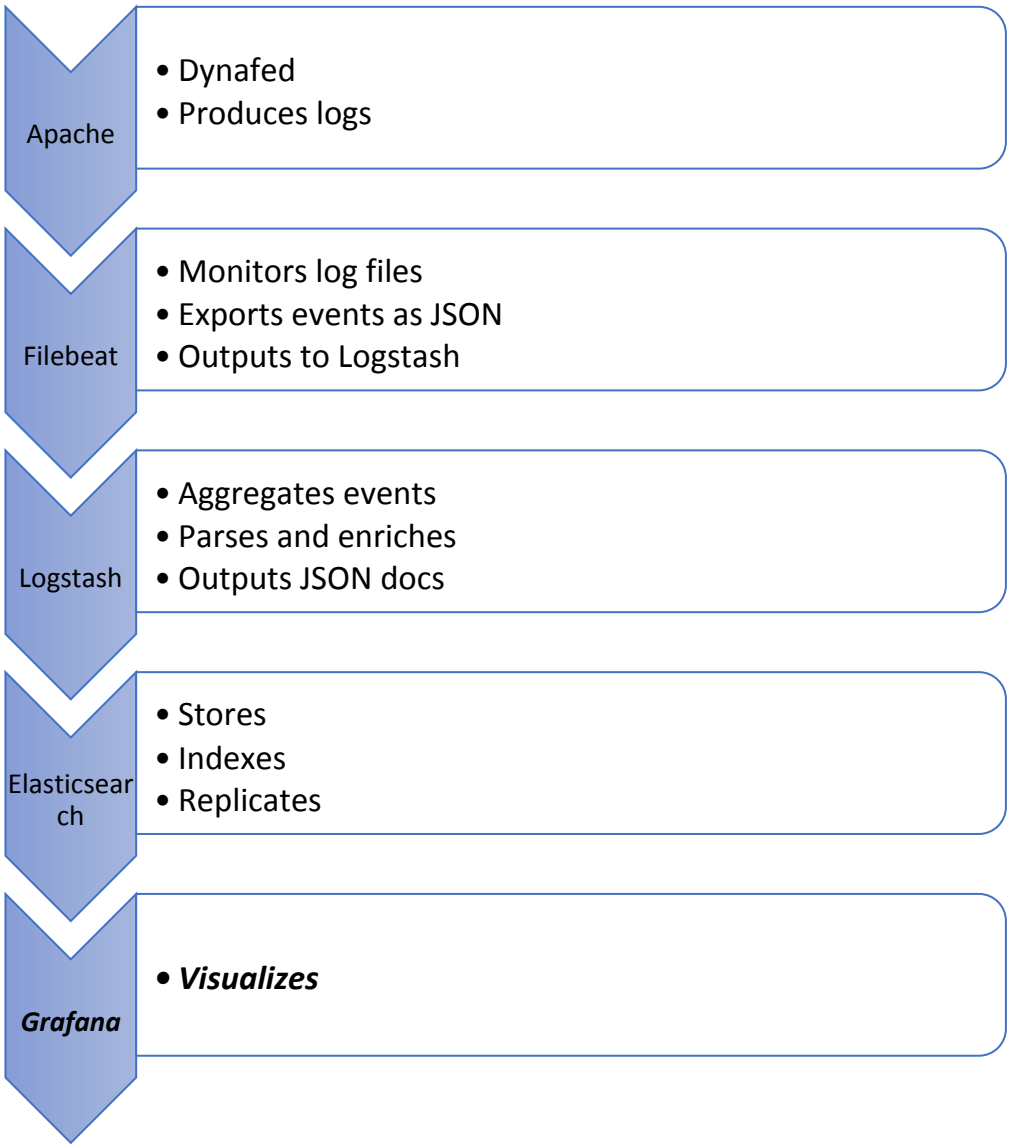
- Events ars stored under two main indices:
  - logstash-dynafed-httpd-acct-*
  - logstash-dynafed-httpd-ops-*
- Replicates events within nodes in the Elasticsearch cluster for redundancy and parallel querying.
- Indices are rotated daily or every 1G events while maintaining the ability to analyze all events by utilizing index aliases.



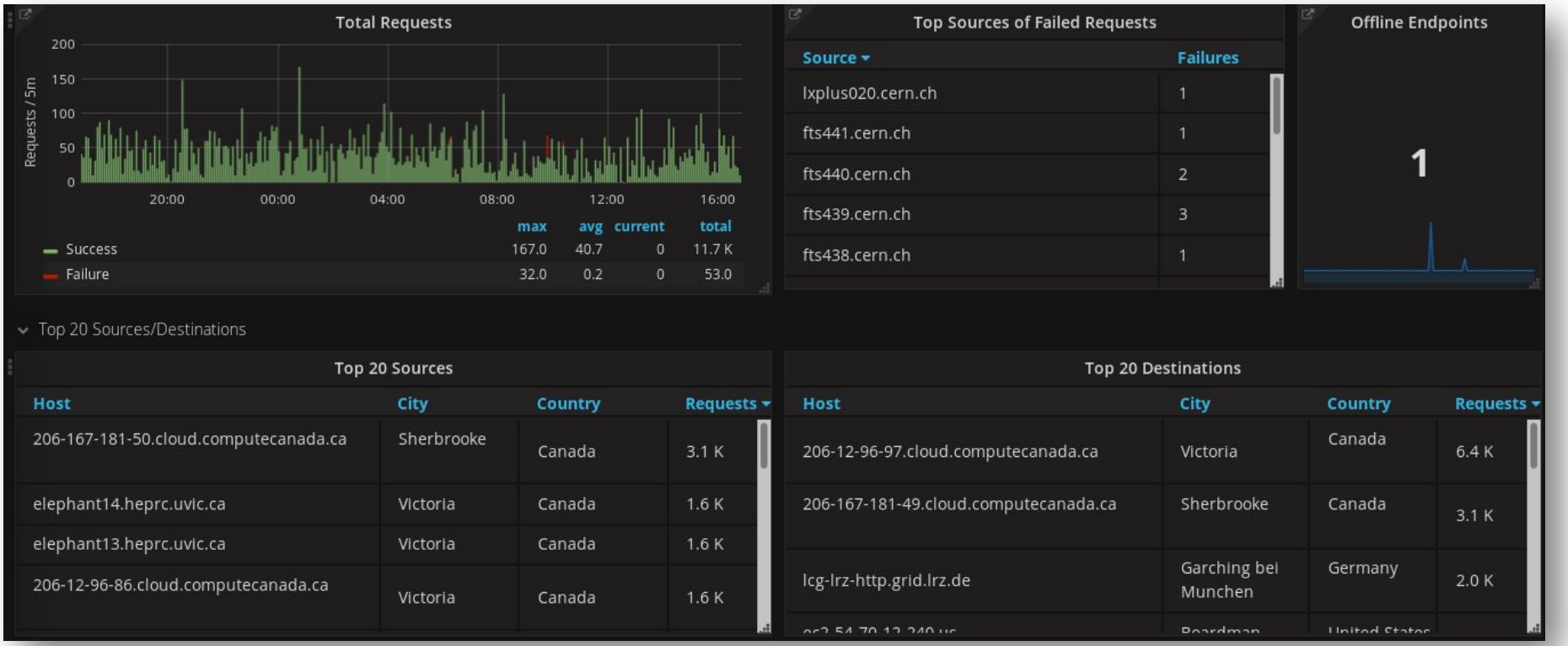
ELK – Stack Process - Grafana

- https://atlas-fed-metrics.triumf.ca/
- Flexibility to create dashboards by querying the Elasticsearch database as well as other sources (like Prometheus).
- Currently provides information per Dynafed host on:
  - Count of success/failure of requests.
  - Count of type of request.
  - Top clients with failed requests.
  - Top 20 sources and destinations SE's.

- Extending the idea with Metricbeats and Execbeats it can also provide:
  - Latency and connection errors to SE's.
  - Apache metrics.



ELK – Grafana – Main Dashboard



ELK – Grafana – Detailed View of Requests