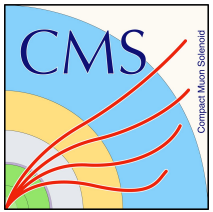
Rucio Experience - ATLAS & CMS

Eric Vaandering (FNAL)
Mario Lassnig (CERN)

5th Rucio Community Workshop
10 - 11 November 2022, Lancaster UK
https://indico.cern.ch/event/1185600/

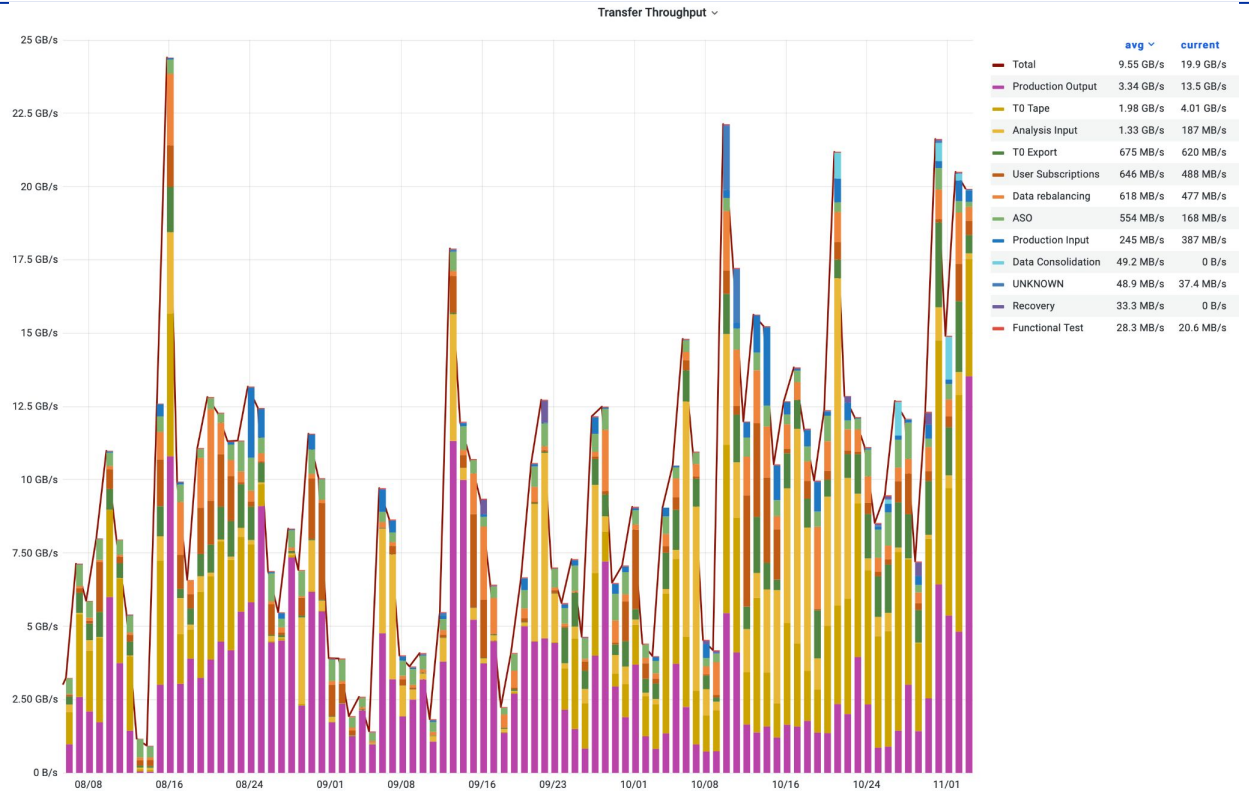


CMS Data Rates

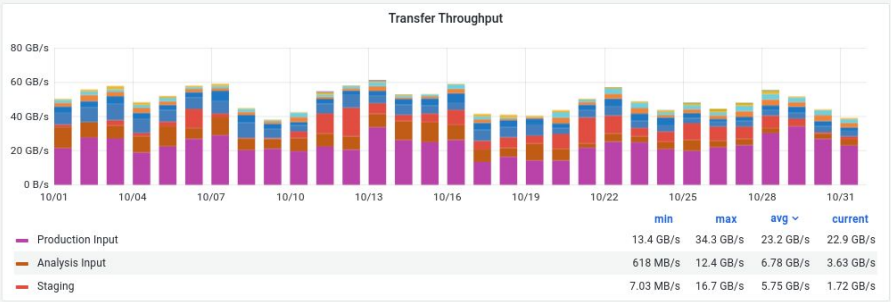
400k transfers/day

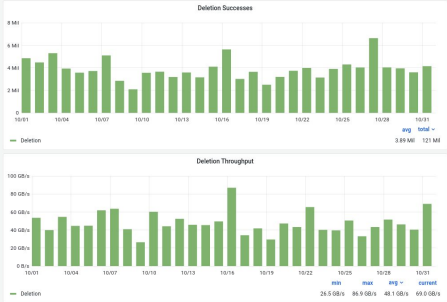
1 PB/day

10 GB/s throughput



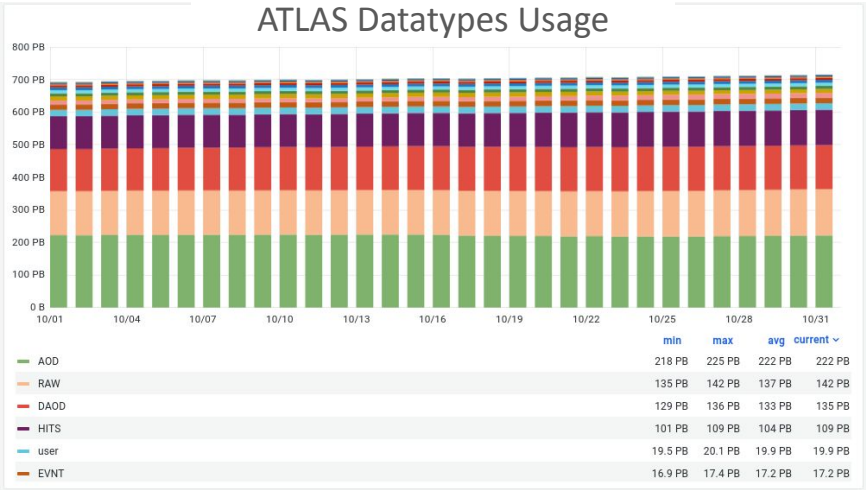
ATLAS Data Rates & Deletion

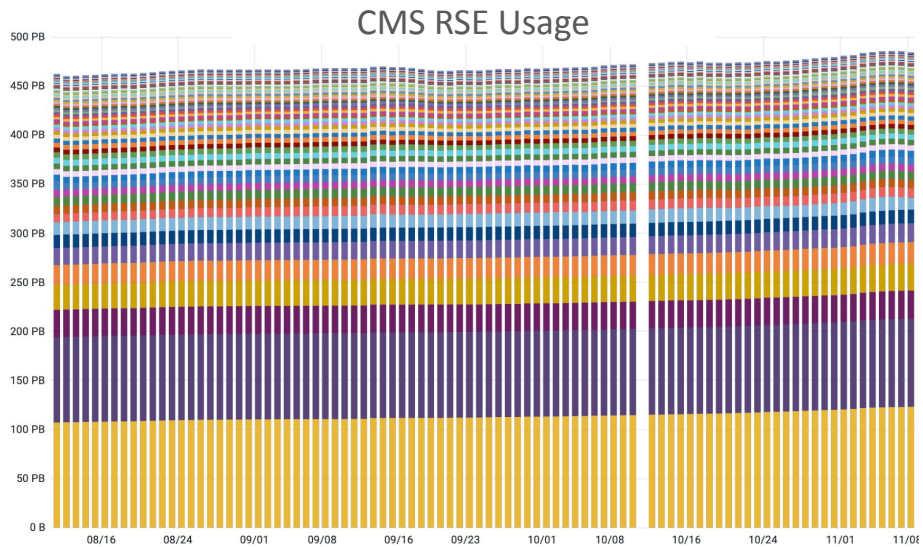




Production (Monte-Carlo Simulation) is dominating ATLAS transfer volume by a factor 3
Deletion volume always slightly surpasses transfer volume
Recent site operations (storage decommissioning, storage migration) are putting significant load on *Data Consolidation* activity
Data Carousel (dynamic exchange of data between disk and tape) is picking up larger fractions of volume at the Tier-1s

Data Volumes





Increase in used space over time is visible already on the small scale (day view)

Dominance of few data types and big data centres clearly visible

Supported by CERN IT monitoring infrastructure for common views and reporting

ATLAS Data Operations

The Daily Struggle ™

The quest for free space

Data categorisation based on rule status

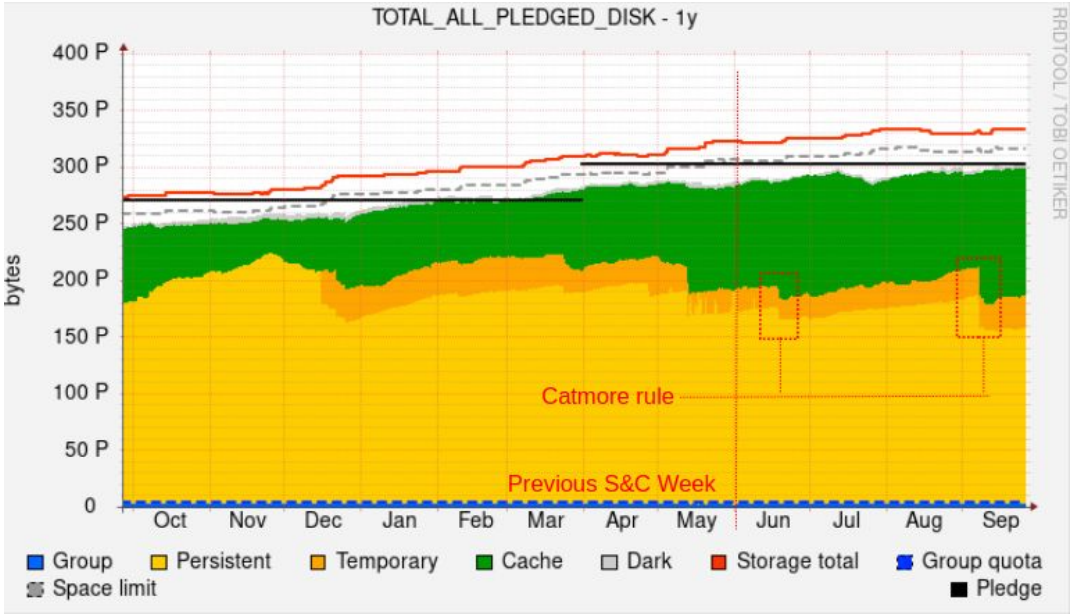
Persistent, Temporary, Cache

Group, Dark

Application of various methods

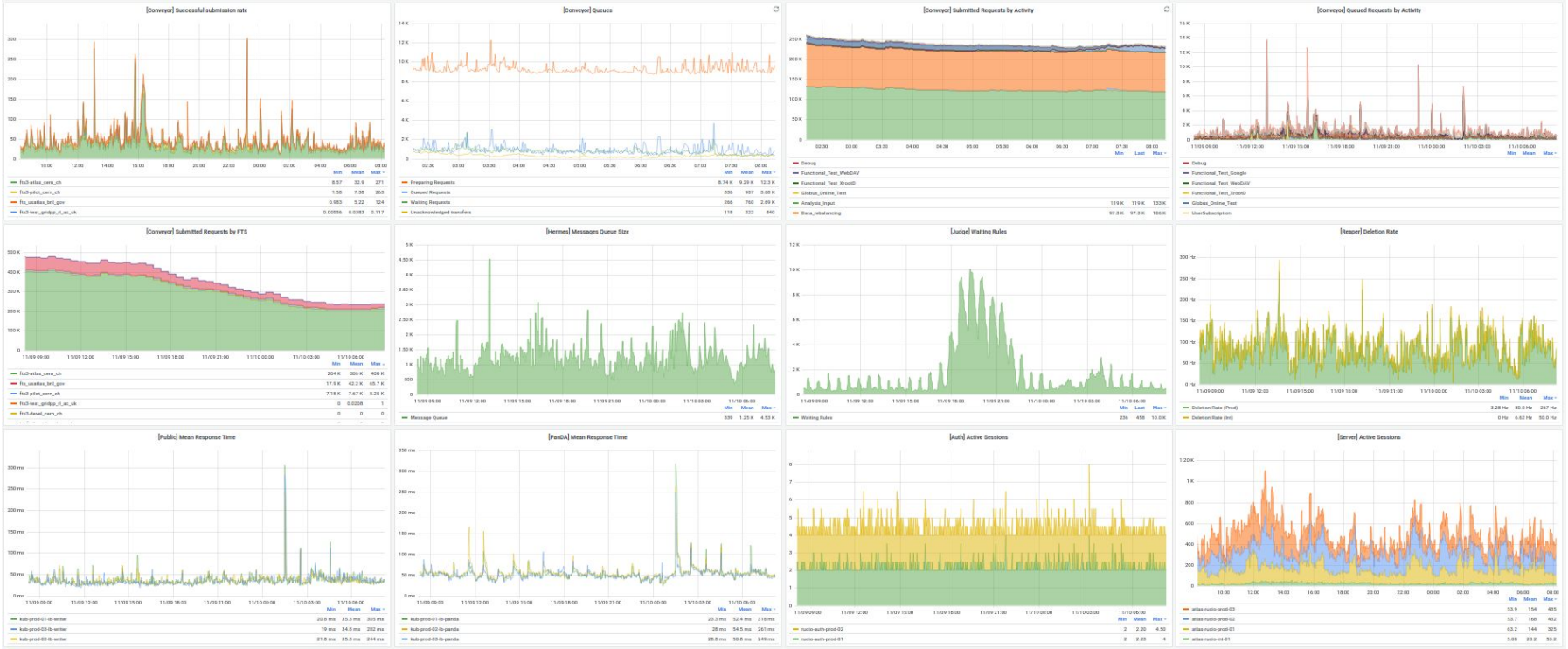
Lifetime model (*delete old&unused*)

Catmore rule (*replica reduction*)



Much healthier available vs total situation than one year ago

Rucio Server Internals Monitoring



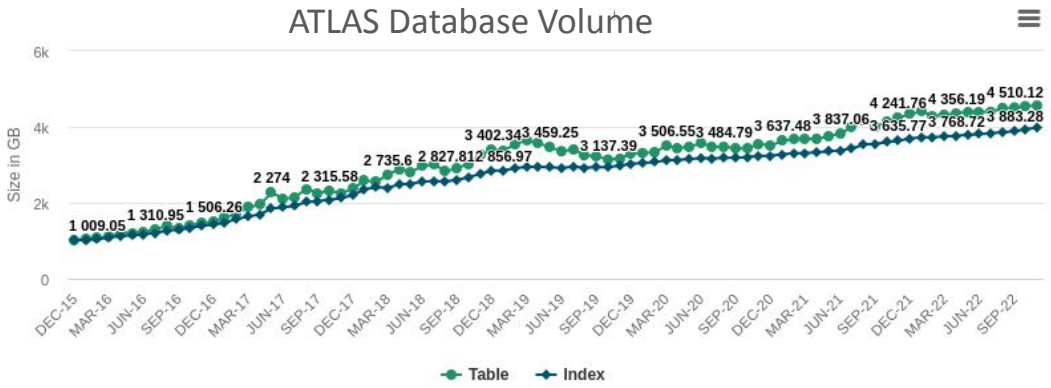
Database Evolution

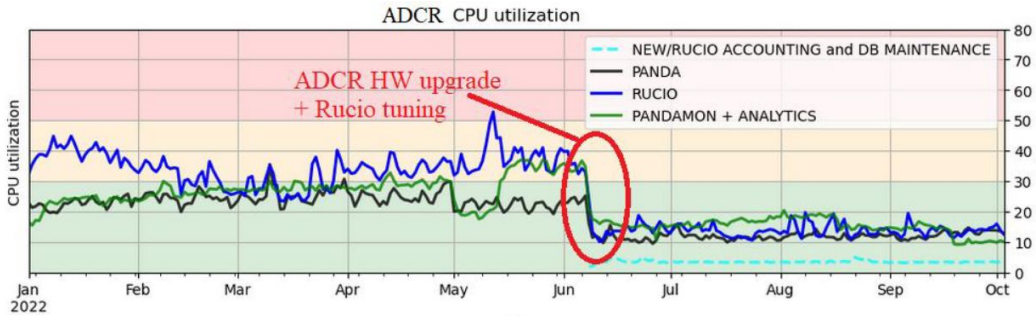
Centrally managed Oracle instance

- Recently upgraded to new hardware and version 19c
- Primary at CERN
- Standby in downtown Geneva

Main points of interest

- Linear database volume growth
- Recent query tuning optimisations
- Offloading of reports to Spark





Kubernetes Setup

ATLAS and CMS setups are very similar

- CMS runs 1 production cluster, ATLAS runs 3 load-balanced production clusters
- Plus 1 integration/testing cluster with one of each kind of server/daemon

Example production cluster setup

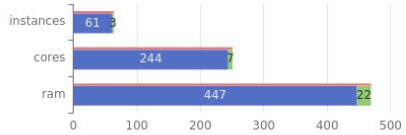
- 6 nodes managed by k8s, helm (application and config manager), flux (GitOps)
- Several of each of the conveyors, reaper, servers
- Otherwise 1-2 of each daemon type
- Dedicated reaper for CERN sites
- Everything Rucio related runs in this cluster
  - Crons for account/site syncing and proxy renewal
  - Probes
  - Consistency enforcement
  - File access collection to generate traces

ATLAS still has a few services running outside Kubernetes from the previous deployment

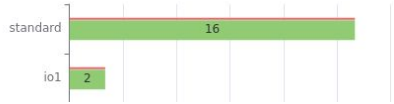
- HAProxy Loadbalancer, Authentication Servers, Synchronisation probes

Kubernetes Setup: OpenStack & GitLab view

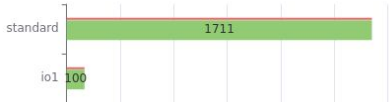
Compute



Number of Volumes



Volume size [GB]



Name	ID	Status	Health Status	Master Count	Node Count
atlas-rucio-int-01	16f40130-4448-452e-a901-15f2922ae793	CREATE_COMPLETE	null	1	5
atlas-rucio-int-02	e1a4fd4e-2cde-43a0-9467-621403b4b705	CREATE_COMPLETE	null	1	4
atlas-rucio-prod-01	7214fbfc-d374-4a69-a268-012432886e58	CREATE_COMPLETE	null	1	14
atlas-rucio-prod-02	0fbf9a62-6100-4dee-88a0-79c940e6d48e	CREATE_COMPLETE	null	1	14
atlas-rucio-prod-03	0322a1ae-6d6e-49bc-b2a9-85d740999230	UPDATE_COMPLETE	null	1	14
atlas-rucio-tools	6513185f-e638-4783-aeb0-a5838f840aef	CREATE_COMPLETE	null	1	2

atlas-adc-ddm
Group ID: 27066 Leave group

ATLAS Distributed Data Management

Subgroups and projects | Shared projects | Archived projects

- M **miscellaneous**
- **flux-rucio**
  Flux configuration for the ATLAS Rucio Kubernetes deployment
- **rpg**
  Replication Policy on the Grid
- F **flux-tools-rucio**
- A **Atlas Rucio Tools**
  Deployment configuration for atlas-rucio-tools cluster
- **storage-monitor**
  Tables and plots of RSE occupancy
- **ddm-ops-toolbox**
  Scripts, SQL queries, and other resources for DDM Ops
- R **rucio-k8s-setup**
  Collection of commands and scripts to install and configure ATLAS Rucio Kubernetes instances
- **rucio-build**
  Rucio tarball build script
- **lifetime-model-policies**
- **ddm-ops-handbook**
  Documentation for DDM Ops
- A **atlas-rucio-logger-prod**
  Configuration for the logging nodes required for Kube

Kubernetes Infrastructure & Logging

3rd party infrastructure (also helm & flux)

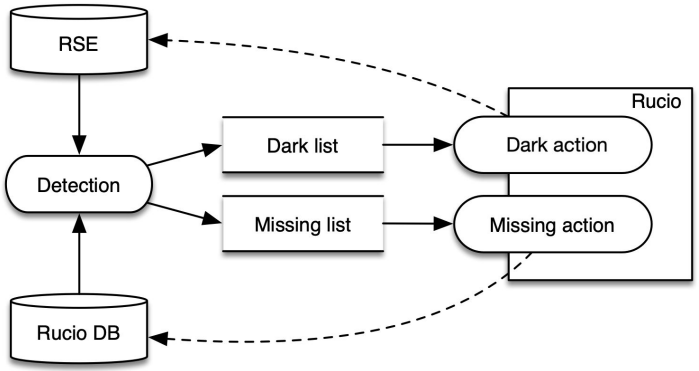
Monitoring via Prometheus

- Prometheus server which scrapes all pods, other components and forwards on to CMS monitoring
- Push gateway for probes and other cron jobs
- statsd-exporter to convert older graphite monitoring to prometheus (including map to labels)

Kube-eagle for pod/node performance monitoring

CERN-provided fluentd for forwarding logs

Consistency Enforcement



Inconsistency detection (simplified):

1. Dump site contents -> list of files on site
2. Dump Rucio DB “replicas” table -> list of expected files
3. Compare (3-way) site contents to DB contents
   a. Dark replicas (not expected)
   b. Missing replicas (expected, not found)

Inconsistency correction actions:

4. Declare missing replicas as BAD
   a. Force Rucio to re-create replicas
5. Quarantine dark replicas
   a. Dark Reaper daemon will remove replicas

Includes relevant conversions between DIDs and physical paths

Consistency Enforcement Toolbox

All sites are scanned using xrootd

- xrdfs ls -l, mostly recursively (-R)
- Exception: RAL (2 RSEs) is not an xrootd server, RSE contents dump done by the admin

Database dump

- Direct access to Rucio database, replicas table via SQLAlchemy

LFN list partitioning then comparison within partitions

```
partition index = hash(LFN) modulo N
```

Rucio ReplicaClient methods to declare replicas missing or quarantine them

Working to share the parts which can be reused by other collaborations

- XRootD scanner
- DB dump
- List partitioning and 3-way comparison
- Client-side code implementing the correction actions algorithms (examples, at least)

CMS Container Monitoring Services

Custom container monitoring in CMS which includes aggregated fields of: size, location(s), last access, creation, event count, replica file count, accessed file count, locked file count, etc.

Data sources: Rucio and DBS (Data Bookkeeping System) tables in OracleDB which are dumped to HDFS in daily schedule.

Aggregation: using Spark jobs. Rucio mainly provides file replicas monitoring in RSEs and DBS provides hierarchical metadata of containers, datasets and files. Joint aggregations of them produces the monitoring data.

Data pipelines: 1-) to MONIT 2-) to MongoDB

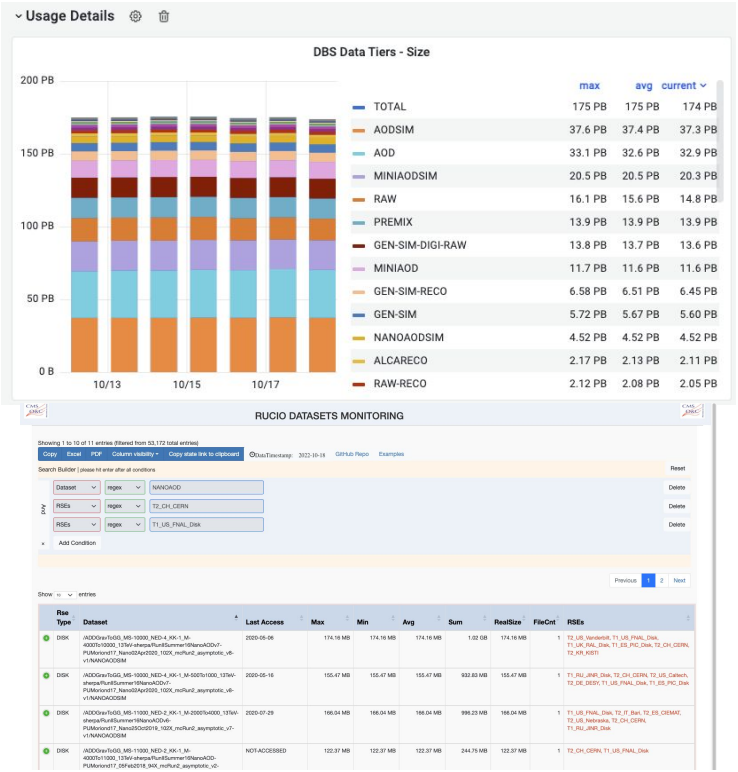
Frontend solutions:
1-) MONIT Kibana&Grafana
2-) JQuery DataTables for tabular view, served by Go web service which also handles REST API for MongoDB queries.

1 Time-series monitoring of datasets

Time-series monitoring data of containers in all RSEs. Accessible through MONIT Kibana and Grafana

2- Tabular datasets monitoring using Go web service and MongoDB

Combination of MongoDB + GoLang web server + JQuery DataTables (with nice hacking of serverside processing via Go) enables tabular view of containers with daily fresh data.



Monitoring Pipeline

Pipeline-1: Daily aggregation results send to MONIT and shown in Grafana dashboards.

Pipeline-2: Daily aggregation results to MongoDB with 1 day of retention time. Powered by fast MongoDB collection management, Golang web server which hosts REST API to MongoDB with all required queries and DataTables with pretty table view which contains complex query builder, details rows, **short url** functionalities.