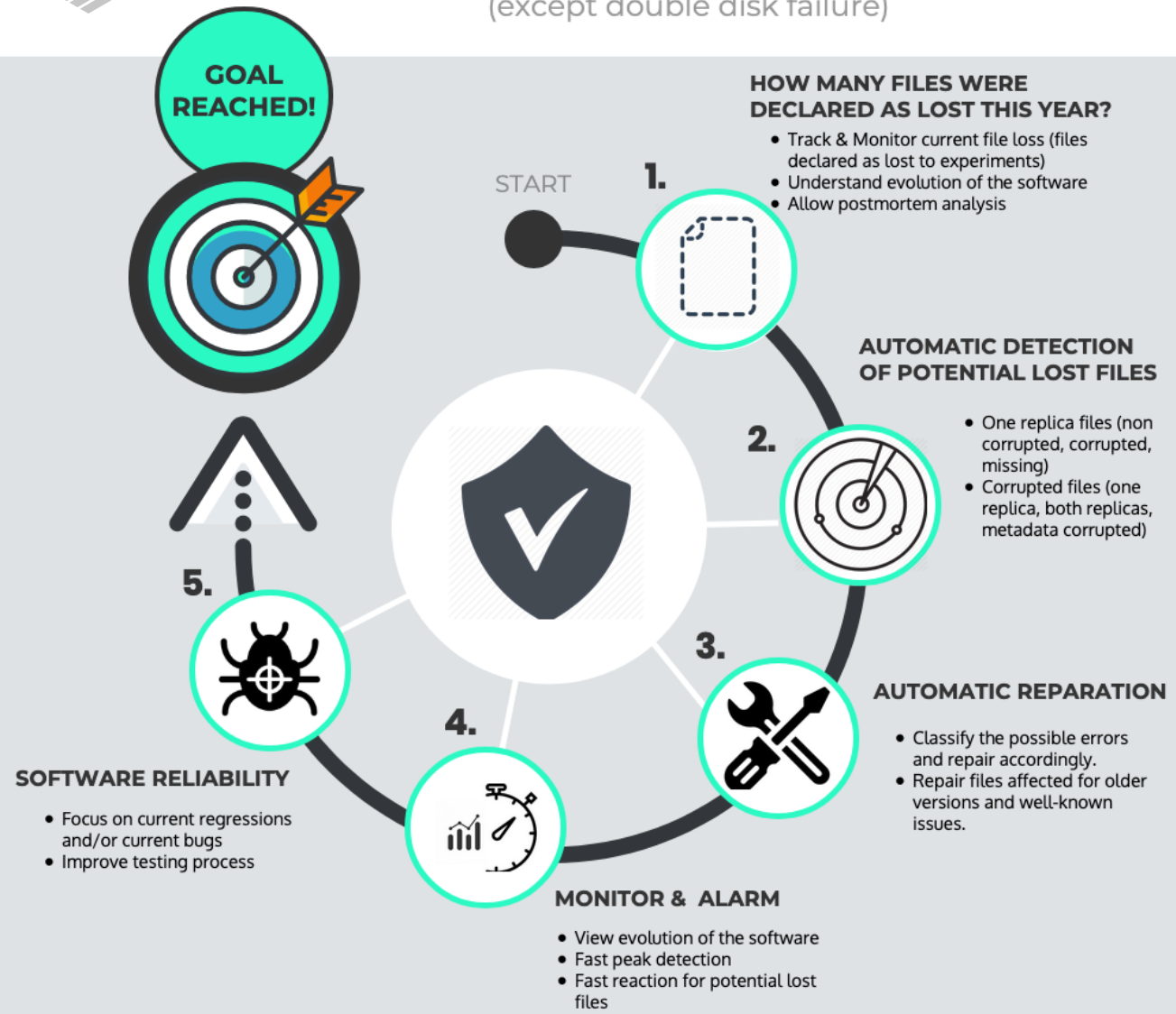
Ensuring Data Durability in EOS Systems

Maria Arsuaga-Rios

IT-ST-PDS

DATA DURABILITY

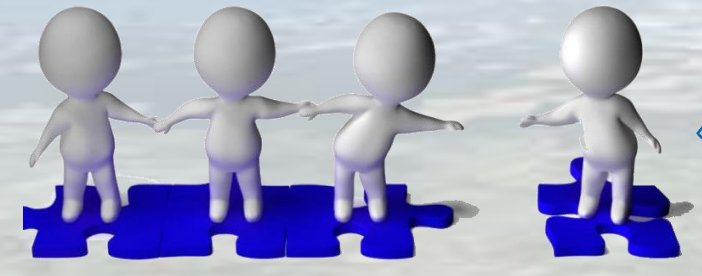
Data is never lost or compromised
(except double disk failure)



EOSPHYSICS: minimal fraction of files may be lost “by design” in certain situations (e.g. double disk failure) — this is a deliberate choice due to the sheer size of the system (100s PBs) and it is trade-off between the cost of storage and required reliability.

EOSUSER: we guarantee better reliability (and we don’t expect the file loss at all) by increasing the cost of the service. This may be done in several ways (e.g. higher replication, additional backup, …).

Flashback: March 2019



Operations support

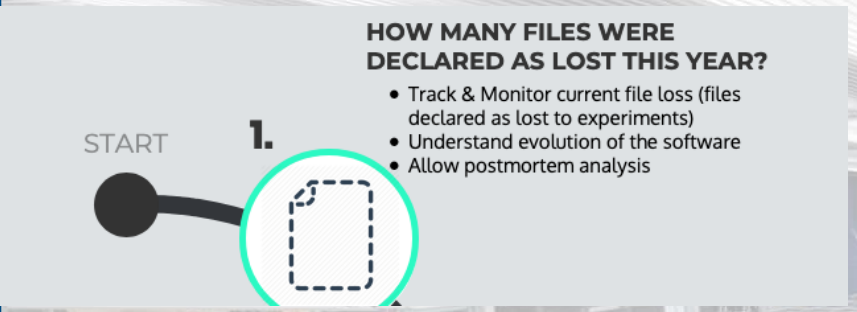
Wigner draining

How many files are we missing?

Is there any tool to allow us post-mortem/historical analysis?

Missing files tracking

- Track and monitor files declared as lost
- Understand evolution of the software
- Allow post-mortem analysis



Lost files Metadata (eos-ops-lostfiles)

elasticsearch

Grafana — Monitoring

kibana — Data discovery

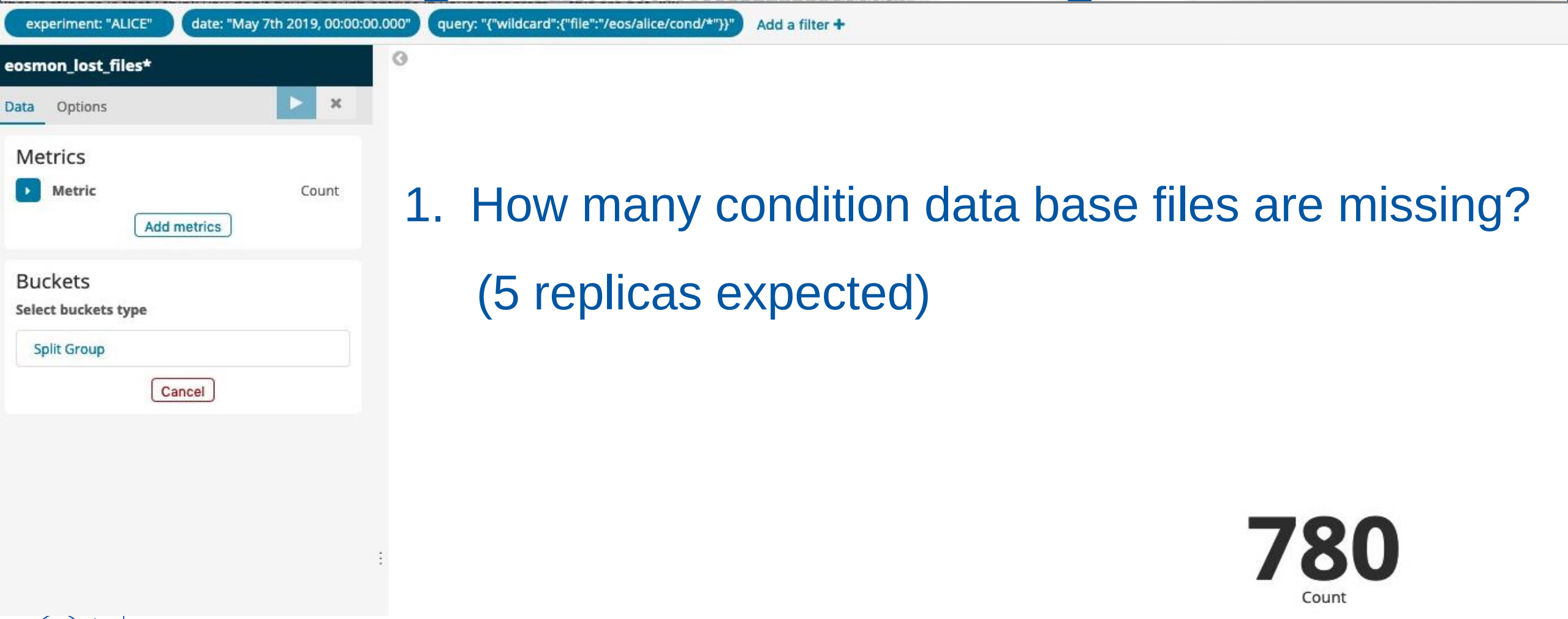
CERNBox

7th May 2019 - we discovered 32K* missing files when draining ALICE

- How we communicate these information to our users?
  1. How many condition data base files are missing?
     - (5 replicas expected)
  2. Which is the distribution over mtime?
  3. Is there a correlation with an incident, ticket or known bug?

*: 0.0046% files stored

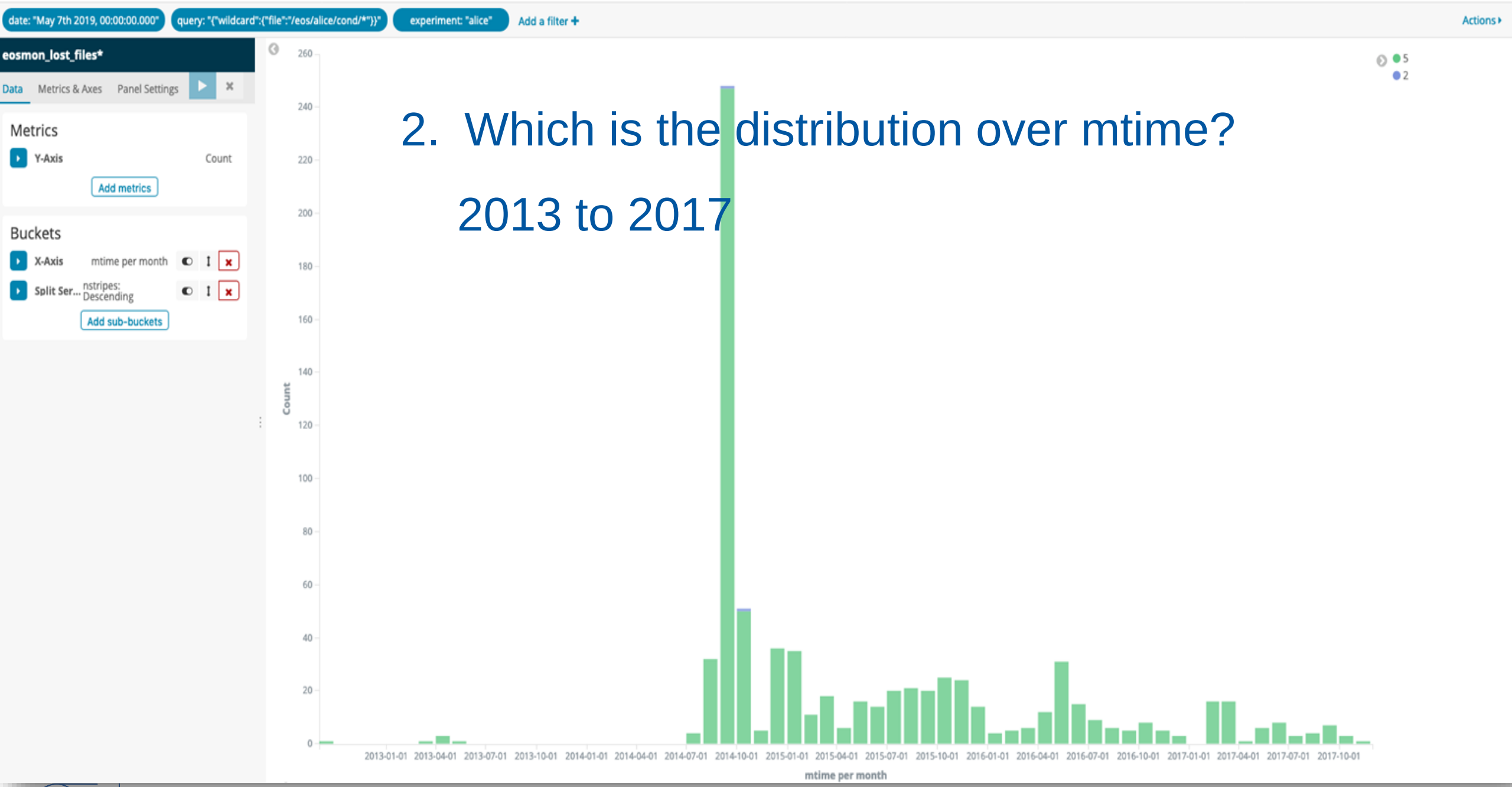
7th May 2019 - we discovered 32K missing files when draining ALICE



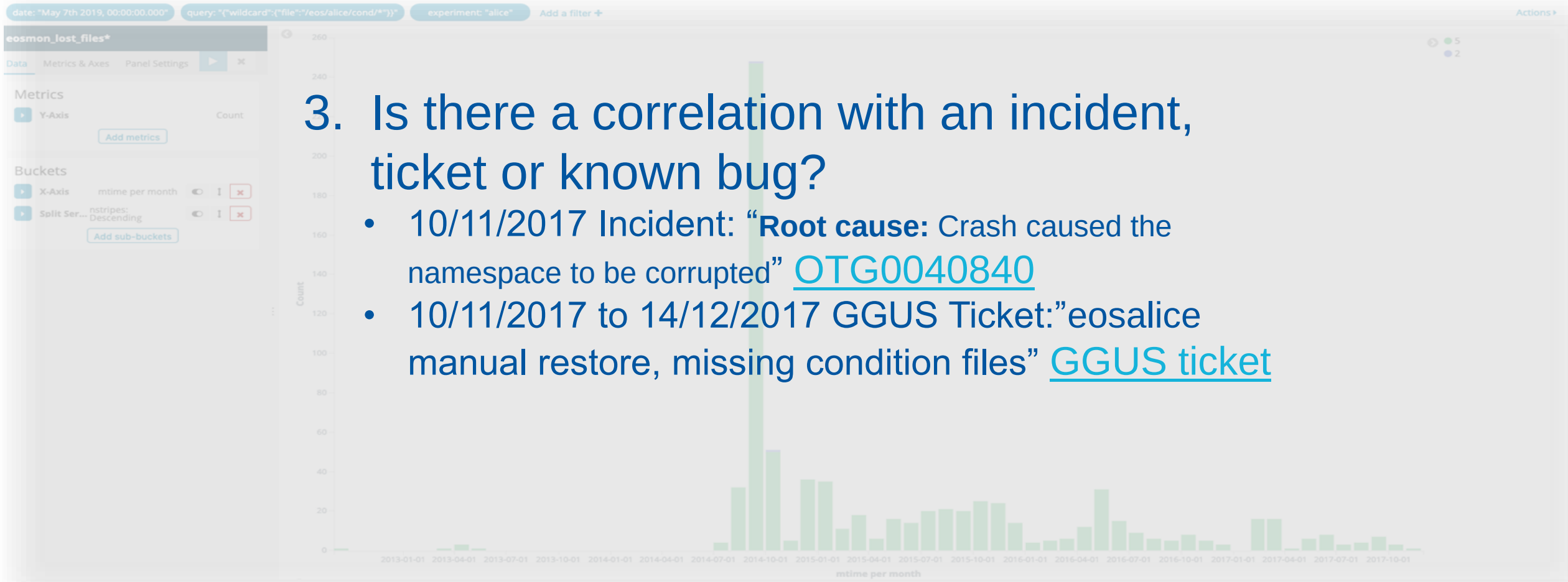
1. How many condition data base files are missing?

   (5 replicas expected)

780
Count

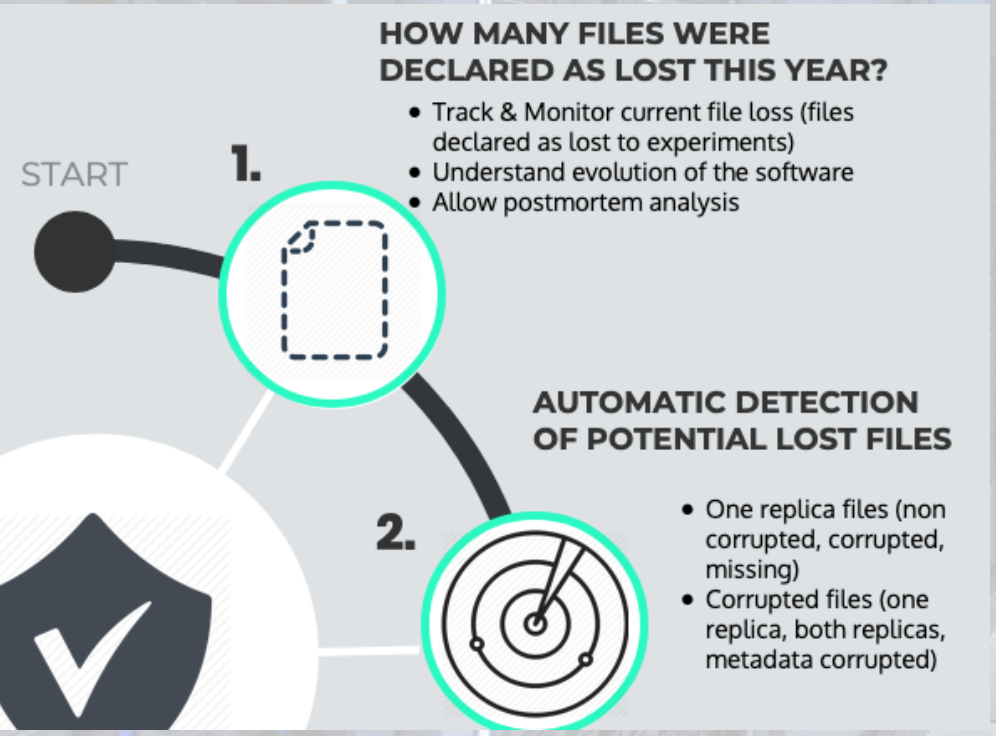

date: "May 7th 2019, 00:00:00.000"
query: "{"wildcard":{"file":"/eos/alice/cond/*"}}"
experiment: "alice"
Add a filter +
Actions
eosmon_lost_files*
Data
Metrics & Axes
Panel Settings
Metrics
Y-Axis
Count
Add metrics
Buckets
X-Axis
mtime per month
Split Ser...
nstripes: Descending
Add sub-buckets
2. Which is the distribution over mtime?
2013 to 2017
Count
mtime per month
2013-01-01
2013-04-01
2013-07-01
2013-10-01
2014-01-01
2014-04-01
2014-07-01
2014-10-01
2015-01-01
2015-04-01
2015-07-01
2015-10-01
2016-01-01
2016-04-01
2016-07-01
2016-10-01
2017-01-01
2017-04-01
2017-07-01
2017-10-01

7th May 2019 - we discovered 32K missing files when draining ALICE



3. Is there a correlation with an incident, ticket or known bug?
   - 10/11/2017 Incident: "**Root cause:** Crash caused the namespace to be corrupted" OTG0040840
   - 10/11/2017 to 14/12/2017 GGUS Ticket:"eosalice manual restore, missing condition files" GGUS ticket

Automatic detection of ***high risk*** files

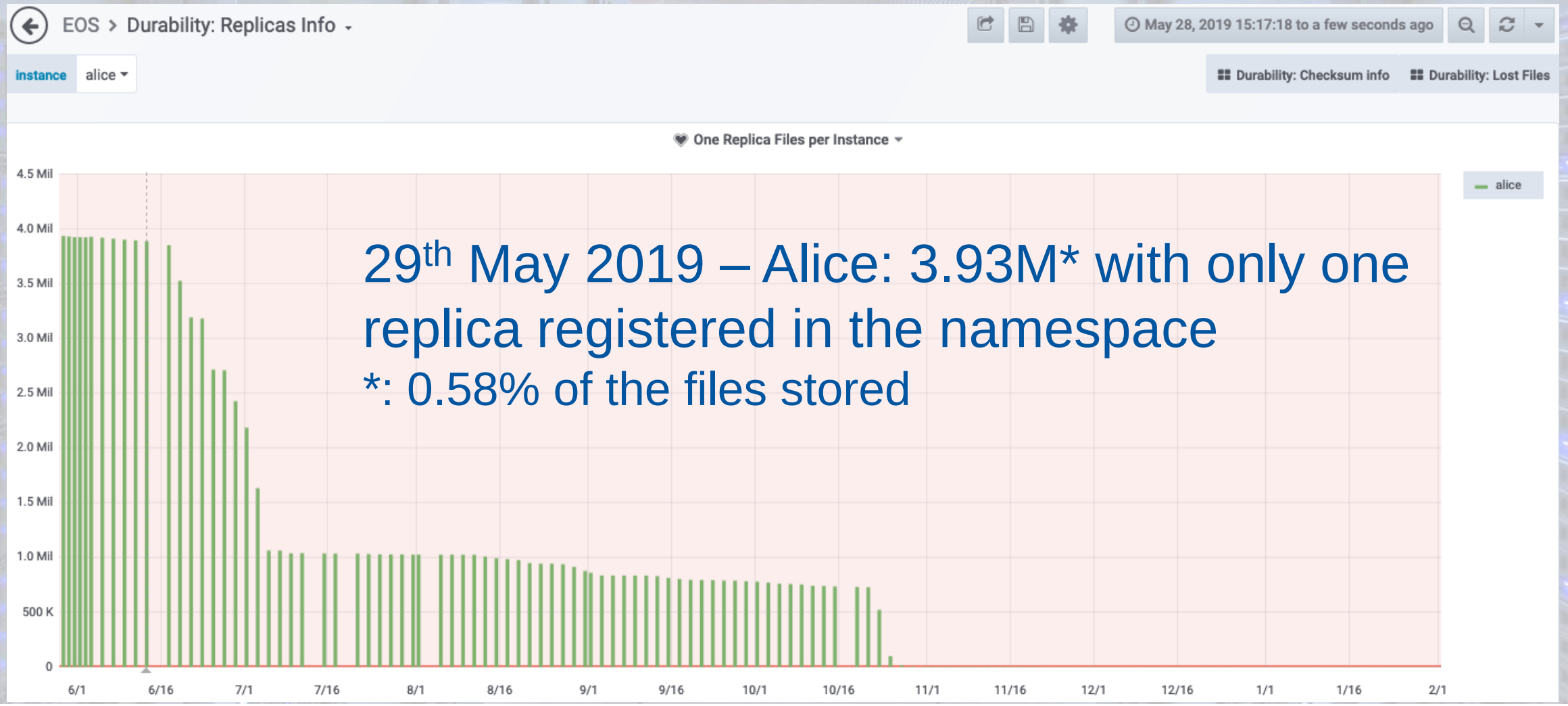


- **Detection is the first step, make it visible!**
  - Detect one replica files (non corrupted, corrupted, missing, …)
  - Detect mismatching checksums and sizes (one replica, both replicas, metadata corrupted, …)

Automatic detection of ***high risk*** files

- One replica files (one replica layouts included):
  - Automatic and daily full scan in all EOS quarkDB instances
    - ✓stripediff option requested to Georgios for the eos-ns-inpect-tool
- Draining failures
  - Automatic and daily full scan in all file systems marked as drained failure (which have problematic files that prevent the completion of the draining process)
- Backup errors
  - Automatic and daily detection of files that couldn’t be backup

Automatic detection of ***high risk*** files

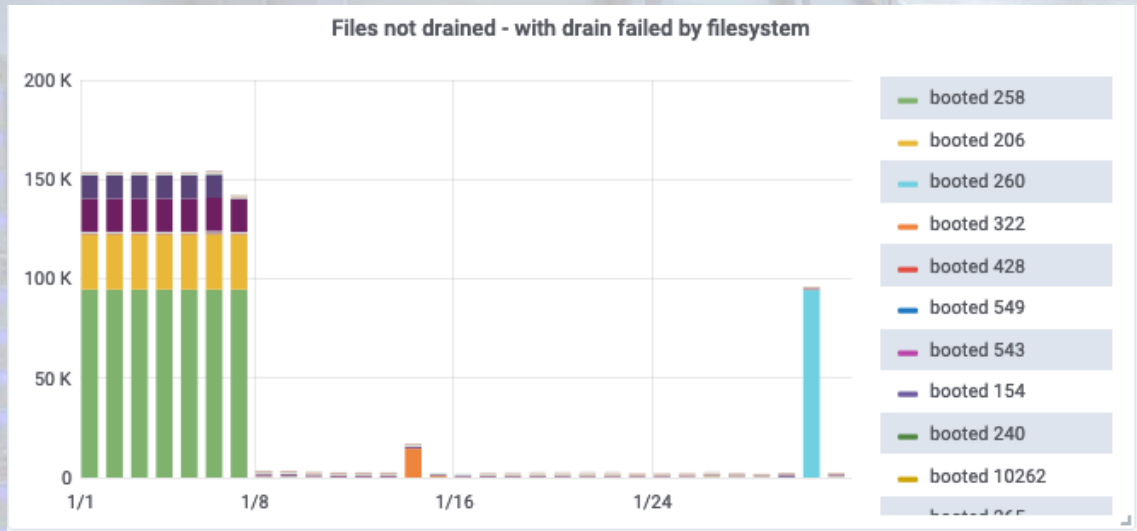


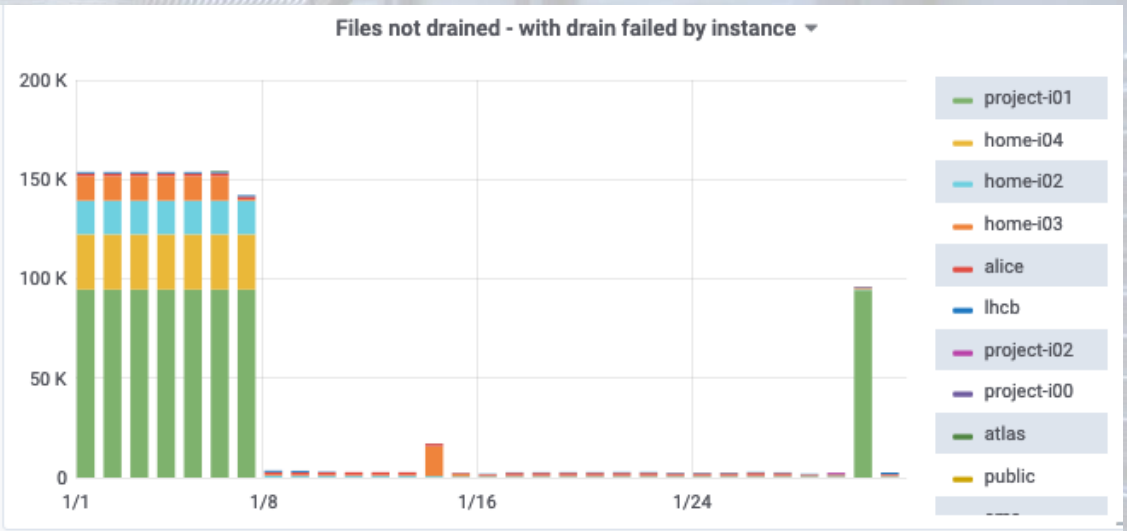
29th May 2019 – Alice: 3.93M* with only one replica registered in the namespace

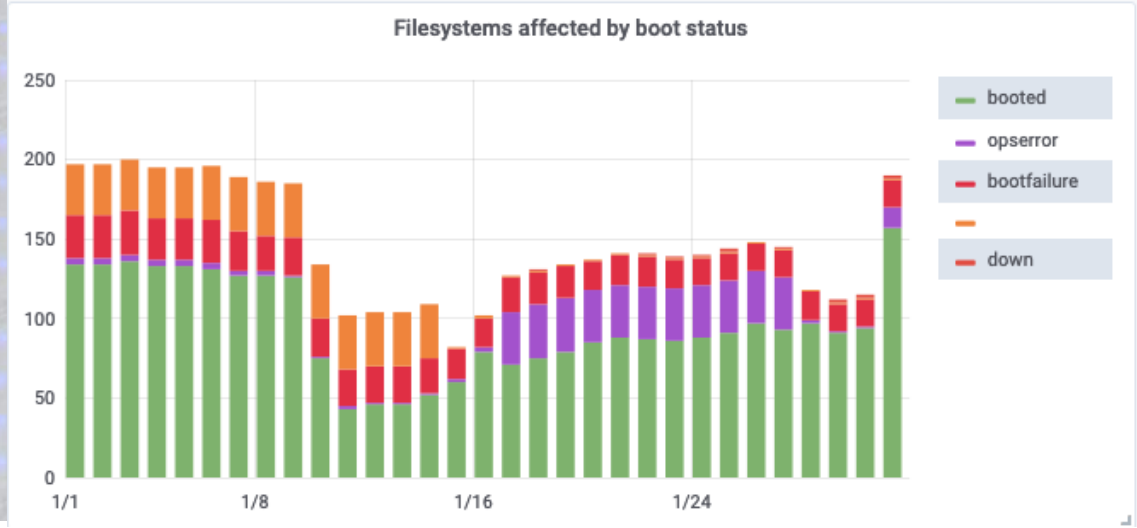
*: 0.58% of the files stored

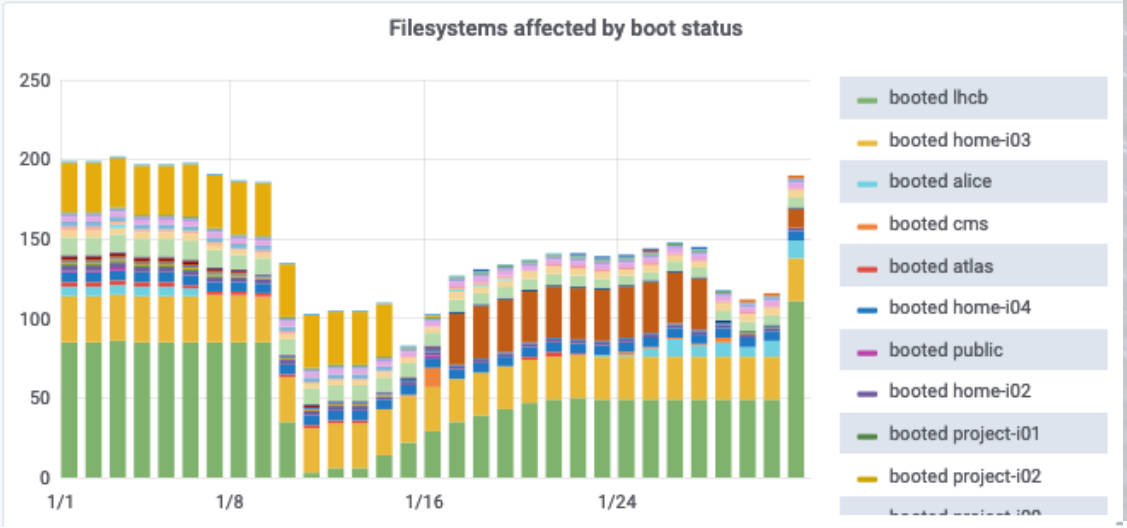
Automatic detection of ***high risk*** files

Draining failures – last month (January 2020)

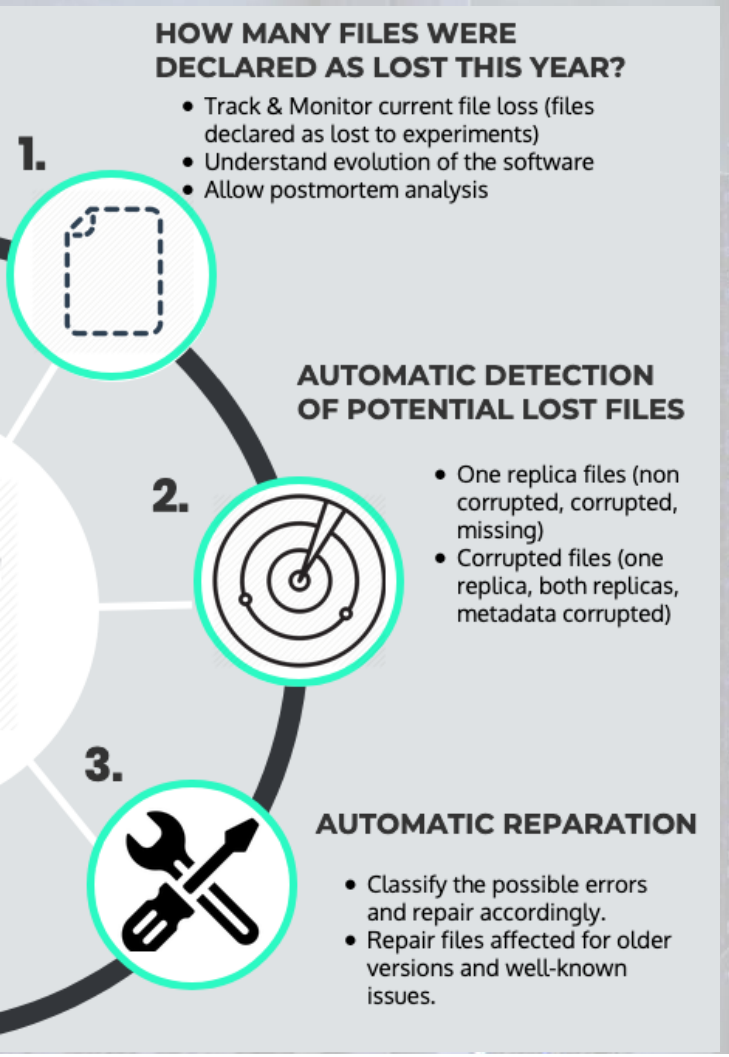






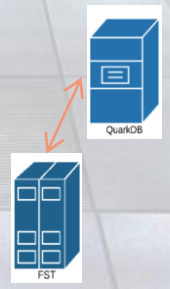


Automatic reparation

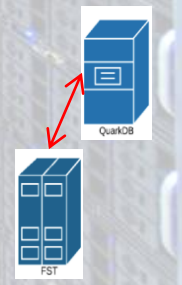


- Namespace full scan is not enough
  - We need to go deeper and get the storage nodes information
- Classify the possible errors and repair accordingly
  - **Divide & Conquer:** 14 categories for one replica files out of 21 categories in total
- Repair files affected for older versions and well-known issues/cases

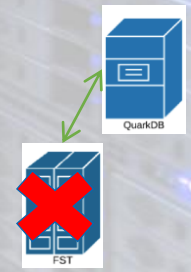
One replica and size and checksum match the namespace



One replica with checksum 0 in namespace



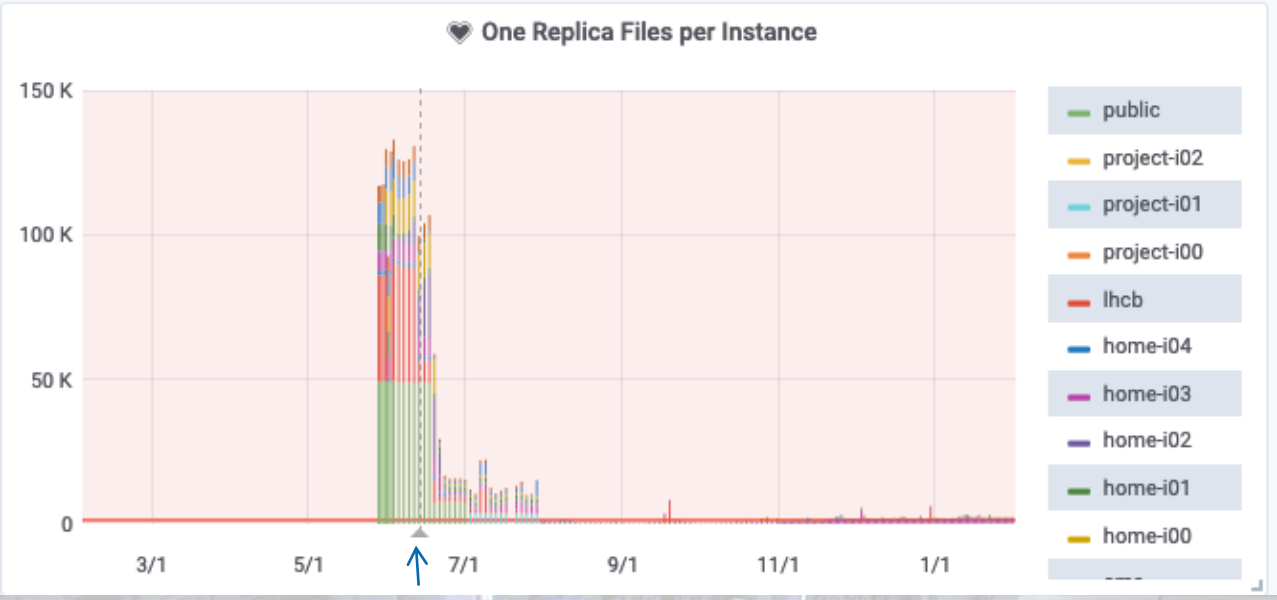
One replica with size and checksum mismatch with namespace

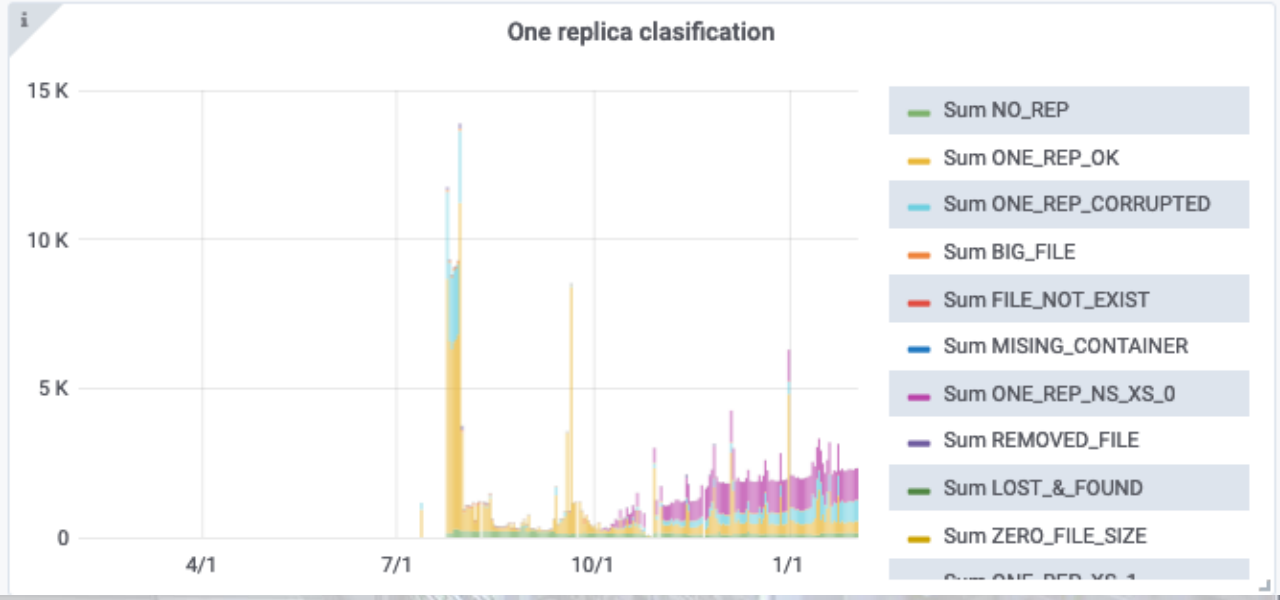


Missing replica

Automatic reparation

- All instances (Alice excluded): One replica files classification and reparation

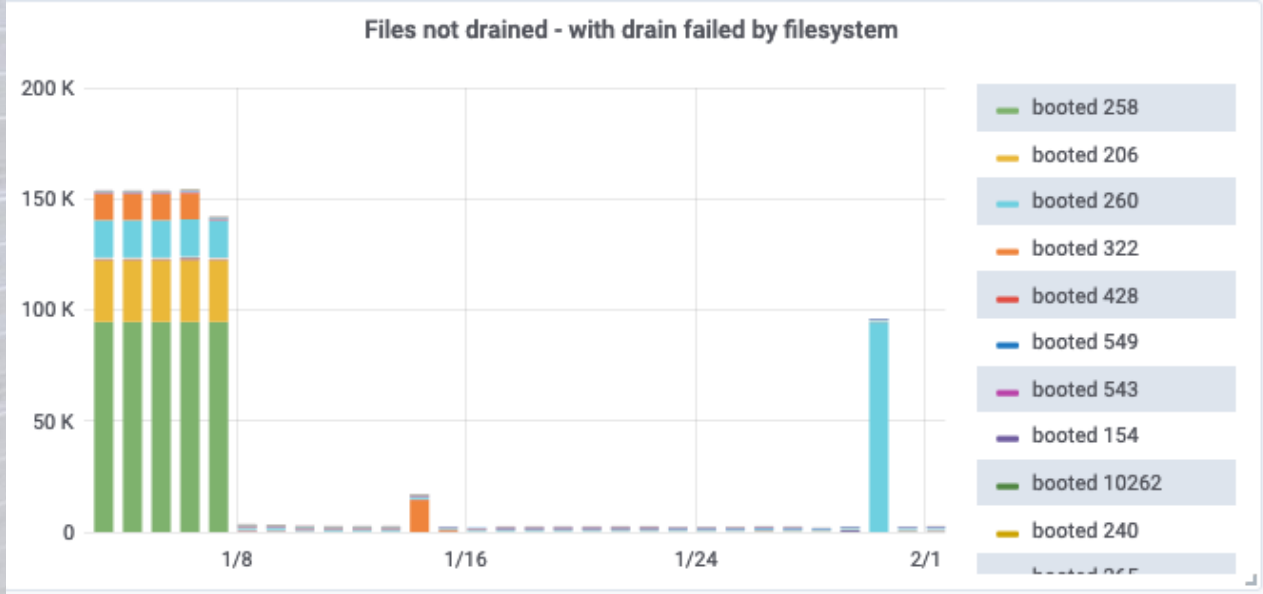


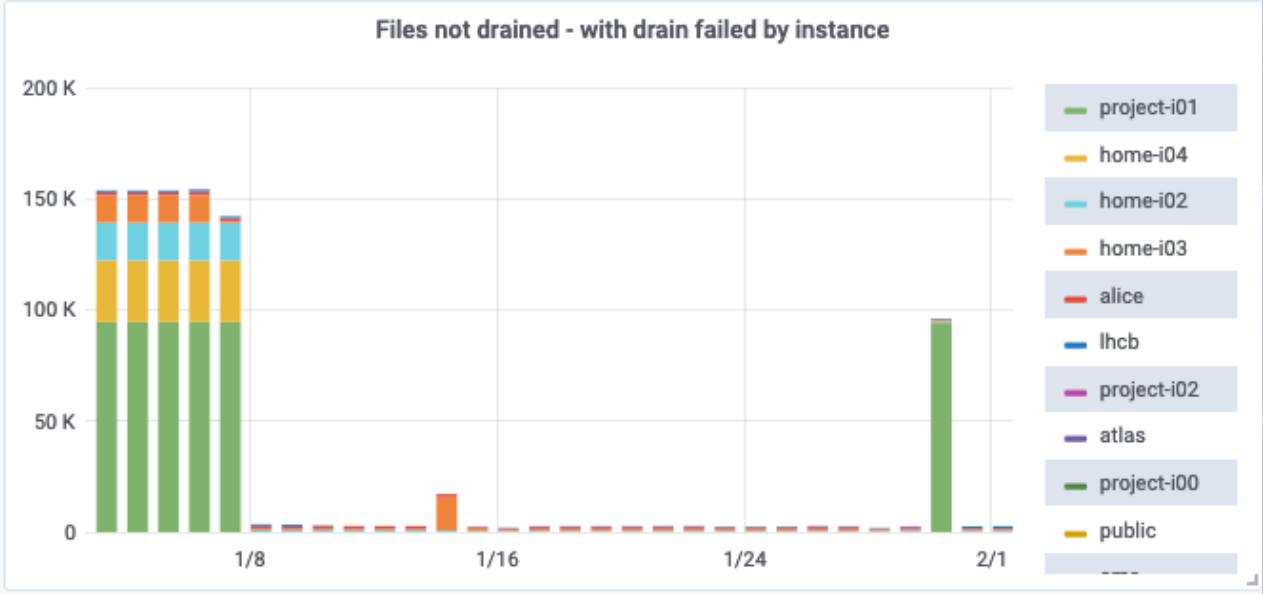


First big reparation because of a massive adjustreplica performed without classification

Automatic reparation

- Automatic reparation for draining failures:
  - Every day at 3pm: ***"Detect + Classify + Repair + Drain" = Less human effort***



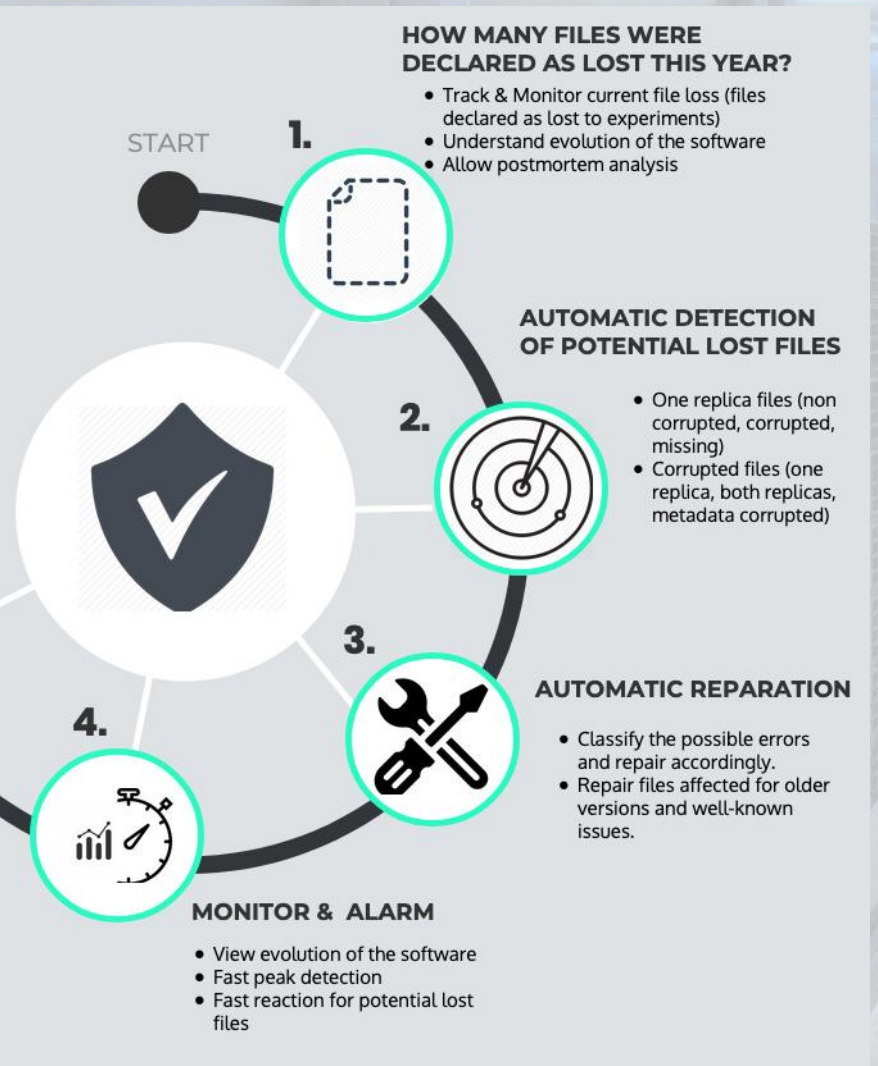


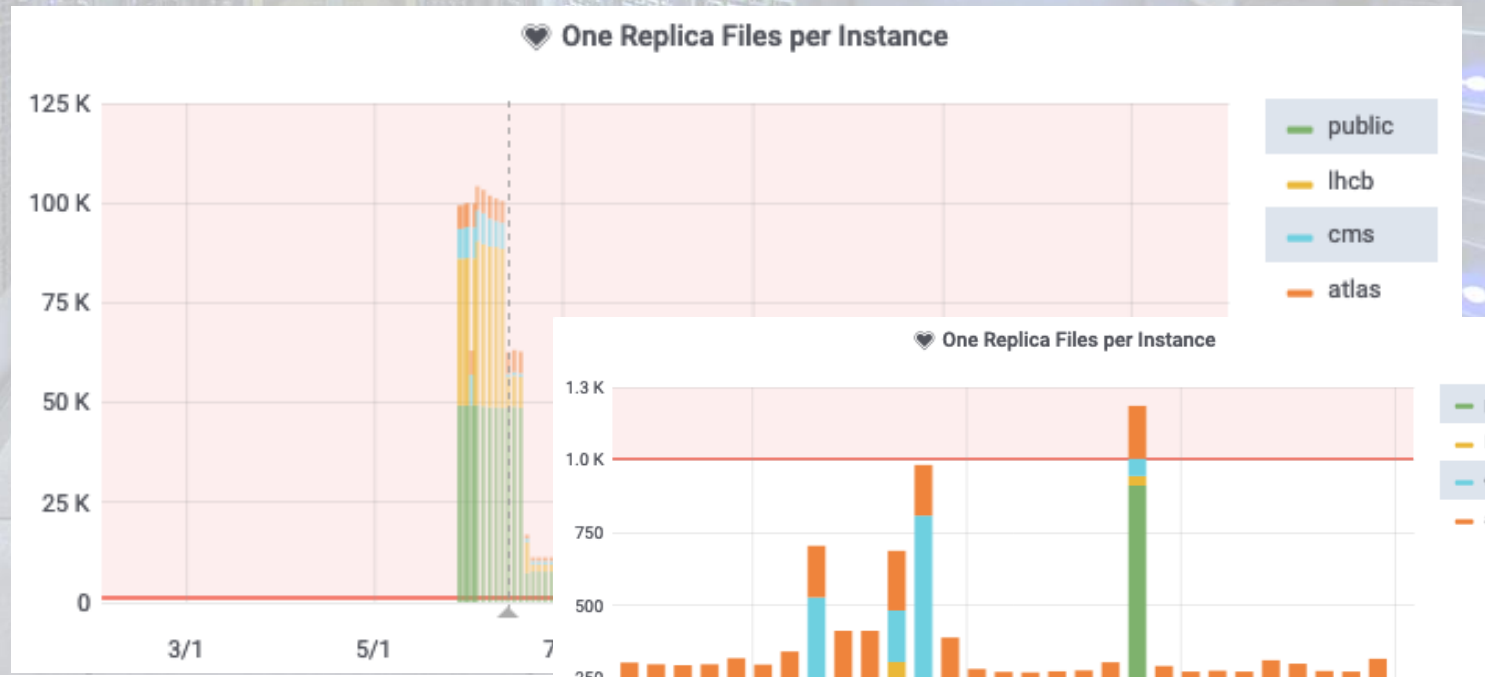
eos-ops-durability toolkit

- Code & automatic rpm creation:
  - ✓Gitlab: https://gitlab.cern.ch/eos/eos-ops-durability
  - ✓EOS repo: http://storage-ci.web.cern.ch/storage-ci/eos-ops-durability
- Installed via puppet in all MGMs (AliceDaq excluded)
- Running in MGMs / rundeck
- Output sent to:
  - Cernbox: https://cernbox.cern.ch/index.php/apps/files/?dir=/__myprojects/eos/Durability&
  - Data source – Elasticsearch + Data Discovery - Kibana: https://es-eosmon.cern.ch/kibana/app/kibana#/discover
  - Monitoring - Grafana: https://filer-carbon.cern.ch/grafana/d/JzDQWU7Zz/durability-classification

Monitor & Alarm

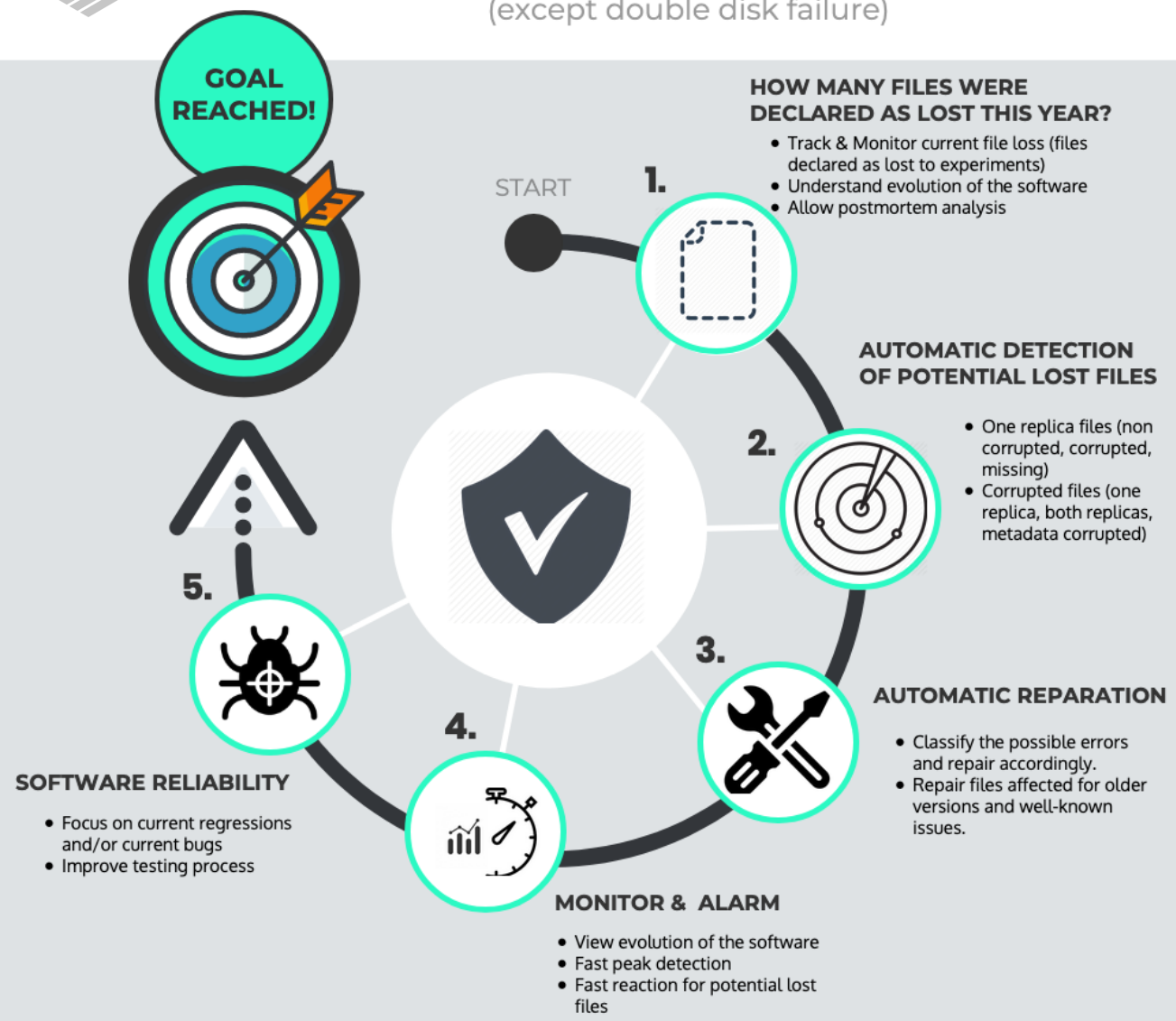
- Allow us to monitor the software evolution
- Fast peak detection and fast reaction





DATA DURABILITY

Data is never lost or compromised
(except double disk failure)



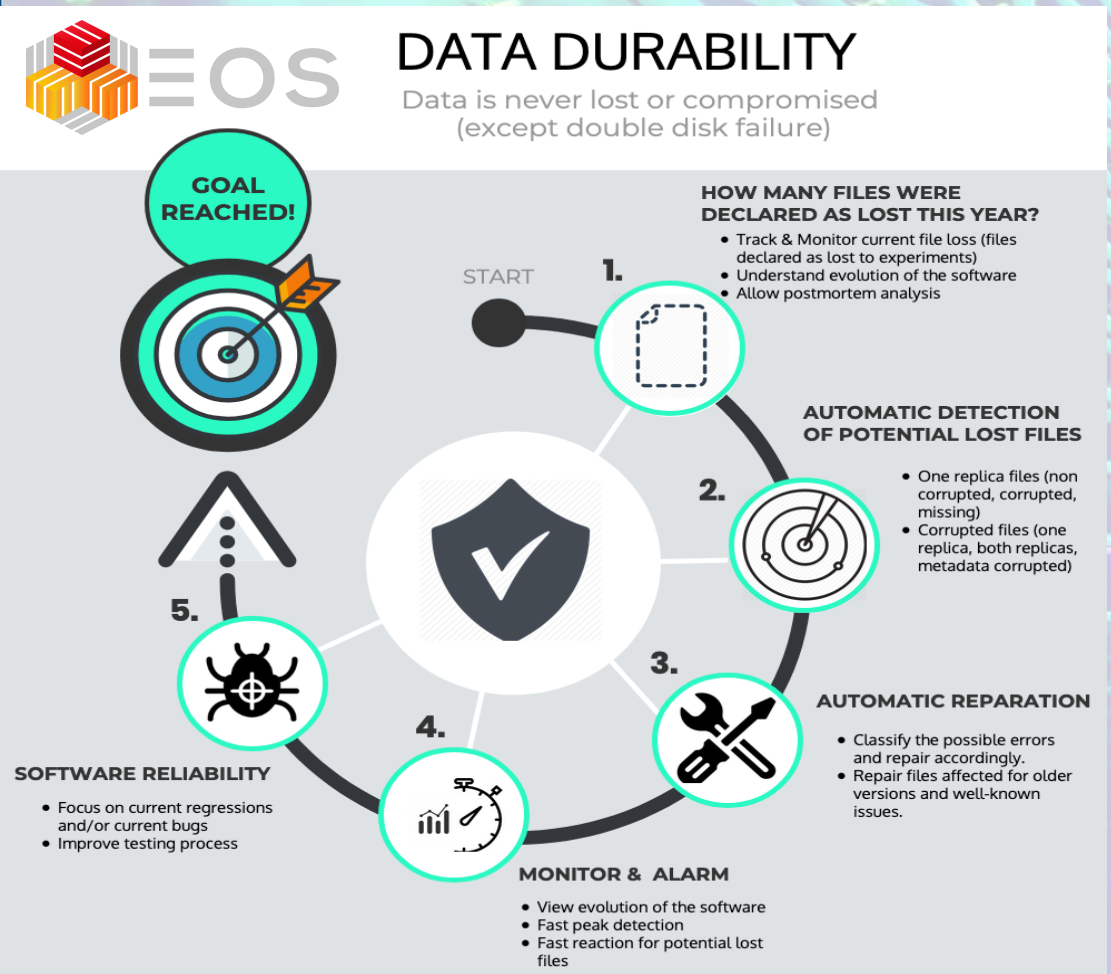
Software reliability

- Focused on current regressions and/or bugs
- Improve testing process
  - Eg. Helping the testing of the new generation of fsck

Future work
Top 4 objectives for next round

- Automatic missing files retrieval from backup and restic in home instances
- Evaluate the new generation of fsck and complement its actions
- Provide external configuration for elasticsearch data sources and eos directories
- Include data durability checks for erasure coding – AliceDaq (complementing fsck)

Conclusions



- Software evolution monitoring
- Better communication and better incidents understanding
- Less human effort in operations support (rota) and draining processes
- Faster peak detection and reaction
- Focus on current regressions/bugs, avoiding noise from the past
- Testing processes improvement