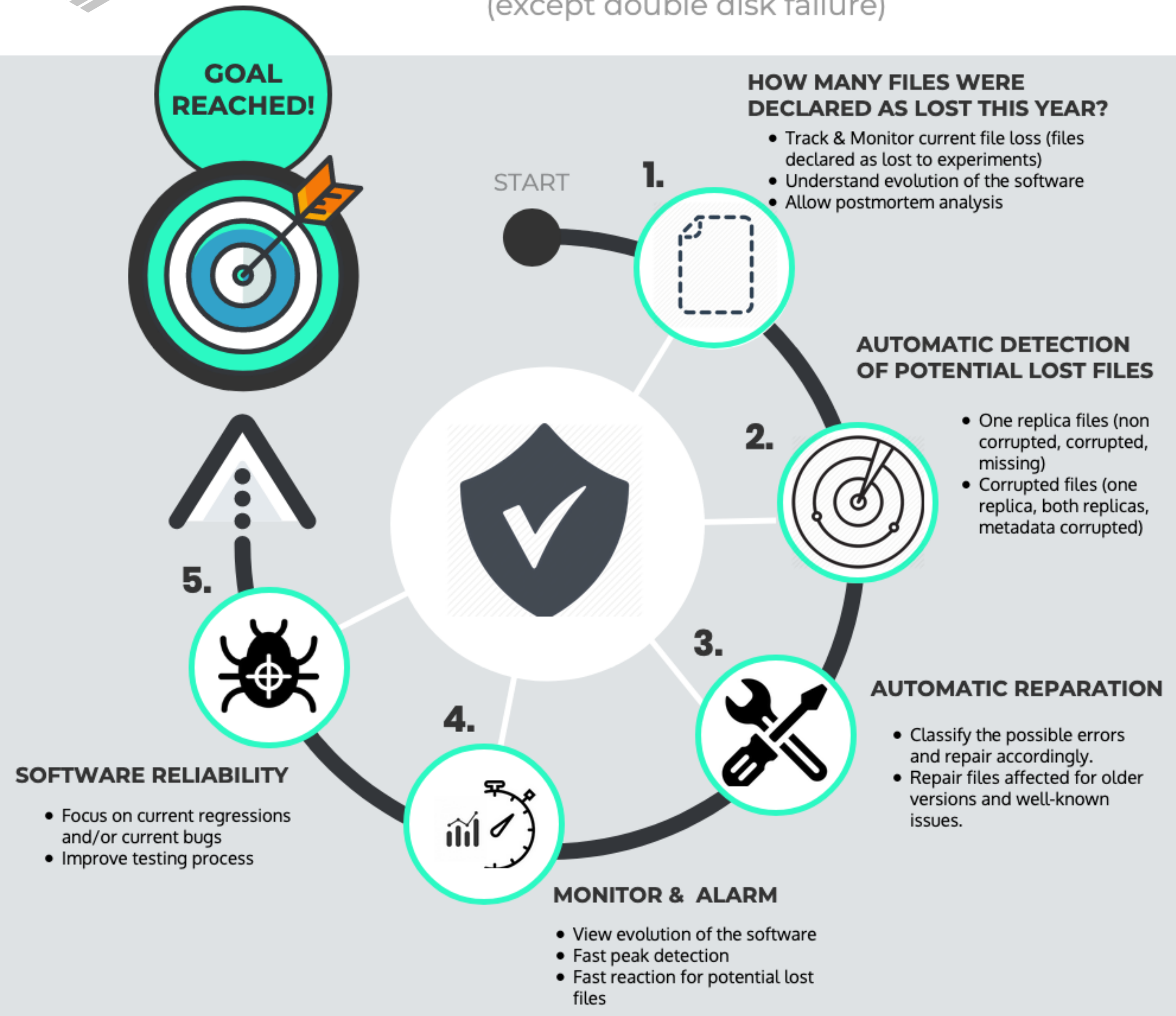
Ensuring Data Durability in EOS Systems

Maria Arsuaga-Rios

IT-ST-PDS

DATA DURABILITY

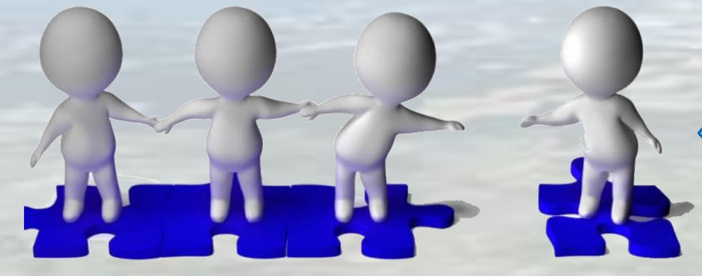
Data is never lost or compromised
(except double disk failure)


GOAL REACHED!
START
1.
HOW MANY FILES WERE DECLARED AS LOST THIS YEAR?
• Track & Monitor current file loss (files declared as lost to experiments)
• Understand evolution of the software
• Allow postmortem analysis
2.
AUTOMATIC DETECTION OF POTENTIAL LOST FILES
• One replica files (non corrupted, corrupted, missing)
• Corrupted files (one replica, both replicas, metadata corrupted)
3.
AUTOMATIC REPARATION
• Classify the possible errors and repair accordingly.
• Repair files affected for older versions and well-known issues.
4.
MONITOR & ALARM
• View evolution of the software
• Fast peak detection
• Fast reaction for potential lost files
5.
SOFTWARE RELIABILITY
• Focus on current regressions and/or current bugs
• Improve testing process

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

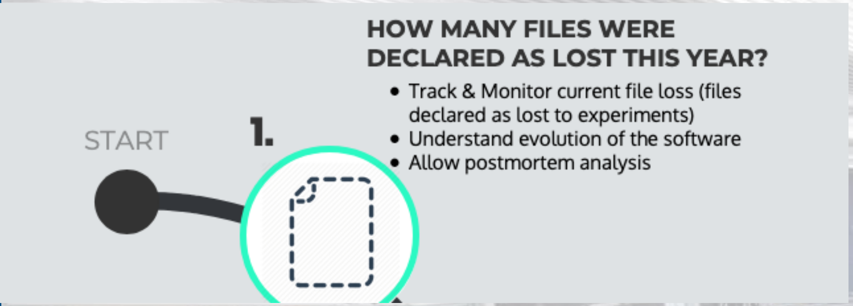
Wigner draining

How many files are we missing?

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

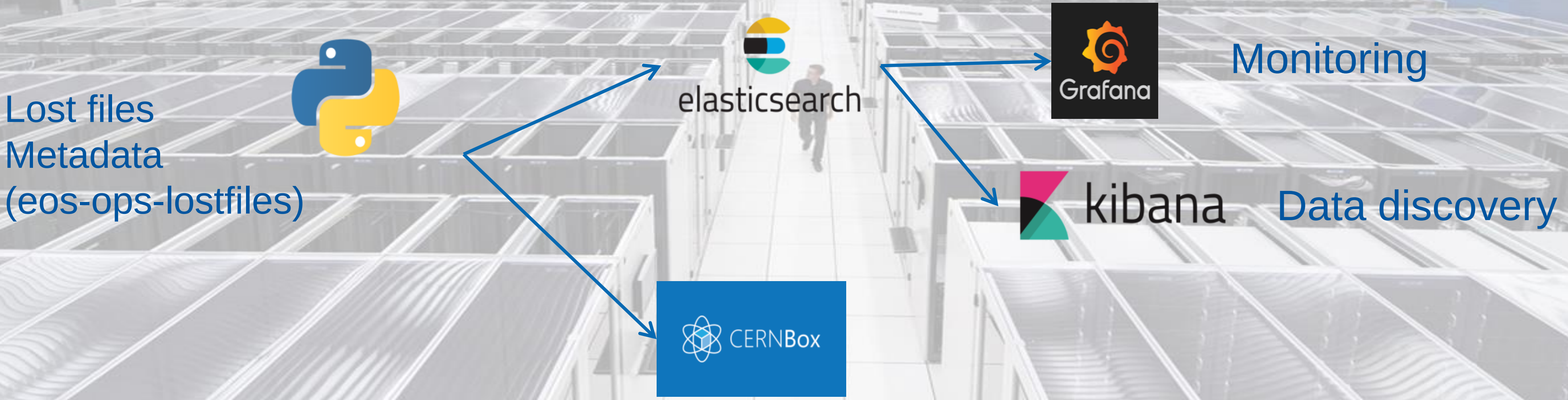
Missing files tracking

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



HOW MANY FILES WERE DECLARED AS LOST THIS YEAR?

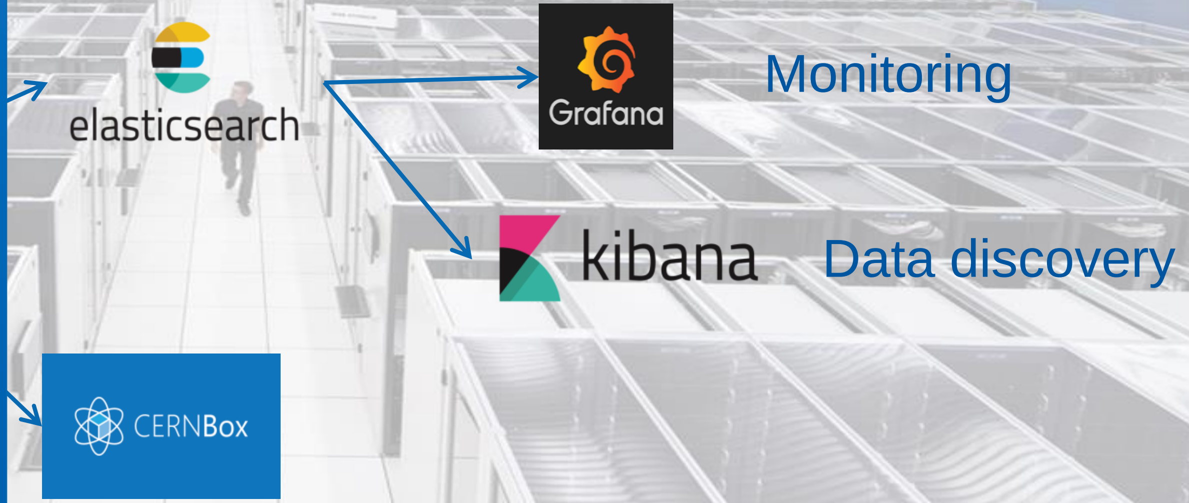
- Track & Monitor current file loss (files declared as lost to experiments)
- Understand evolution of the software
- Allow postmortem analysis



```
eos-ops-lostfiles -f file -d 2020-02-04 -r "bootfailure_with_other_replica_0_size" -e ALICE --send
```

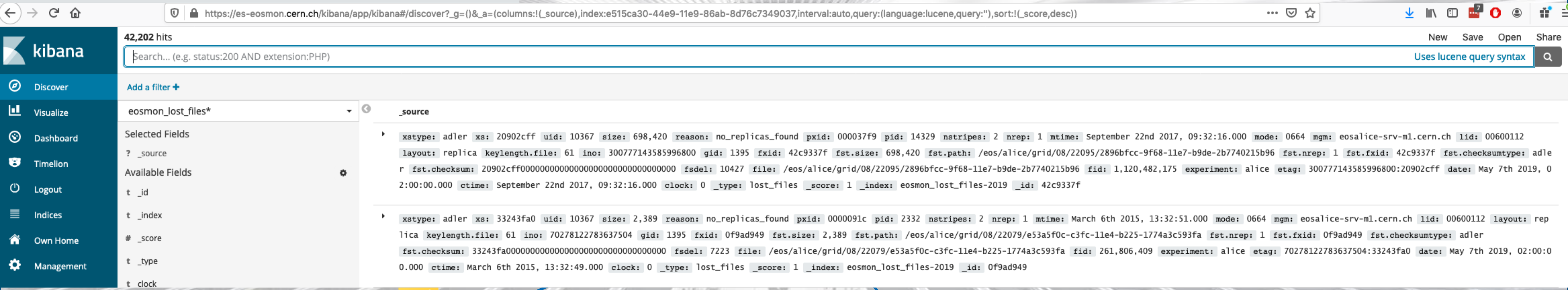
Missing files tracking

```
send_data_to_es(data, index, _id, es):
        requestBody = ""
        requestBody+='{"index":{"_id": "'+_id+'"}}\n'
        requestBody+= json.dumps(data) +"\n”

        try:
                out = es.bulk(index='eosmon_'+index+'-
'+str(datetime.datetime.now().year), doc_type=index, body=requestBody)
                print out
        except TransportError as e:
                print e.info
        except Exception as er:
                print er

get_es_connexion():
        USER_ES = 'eosmon'
                PASSWORD_ES =  lalala
                certpath = "/etc/pki/tls/certs/ca-bundle.trust.crt"
                  es = Elasticsearch(
                'https://'+USER_ES+':'+PASSWORD_ES+'@es-
eosmon.cern.ch/es',
                # turn on SSL
                use_ssl=True,
                # make sure we verify SSL certificates (off by default)
                verify_certs=True,
                ca_certs=certpath)
          return  es
```



Missing files tracking

HOW MANY FILES WERE DECLARED AS LOST THIS YEAR?


https://es-eosmon.cern.ch/kibana/app/kibana#/discover?_g=()&_a=(columns:!(_source),index:e515ca30-44e9-11e9-86ab-8d76c7349037,interval:auto,query:(language:lucene,query:''),sort:!(_score,desc))
kibana
Discover
Visualize
Dashboard
Timelion
Logout
Indices
Own Home
Management
42,202 hits
New
Save
Open
Share
Search... (e.g. status:200 AND extension:PHP)
Uses lucene query syntax
Add a filter +
eosmon_lost_files*
Selected Fields
? _source
Available Fields
t _id
t _index
# _score
t _type
t clock
_source

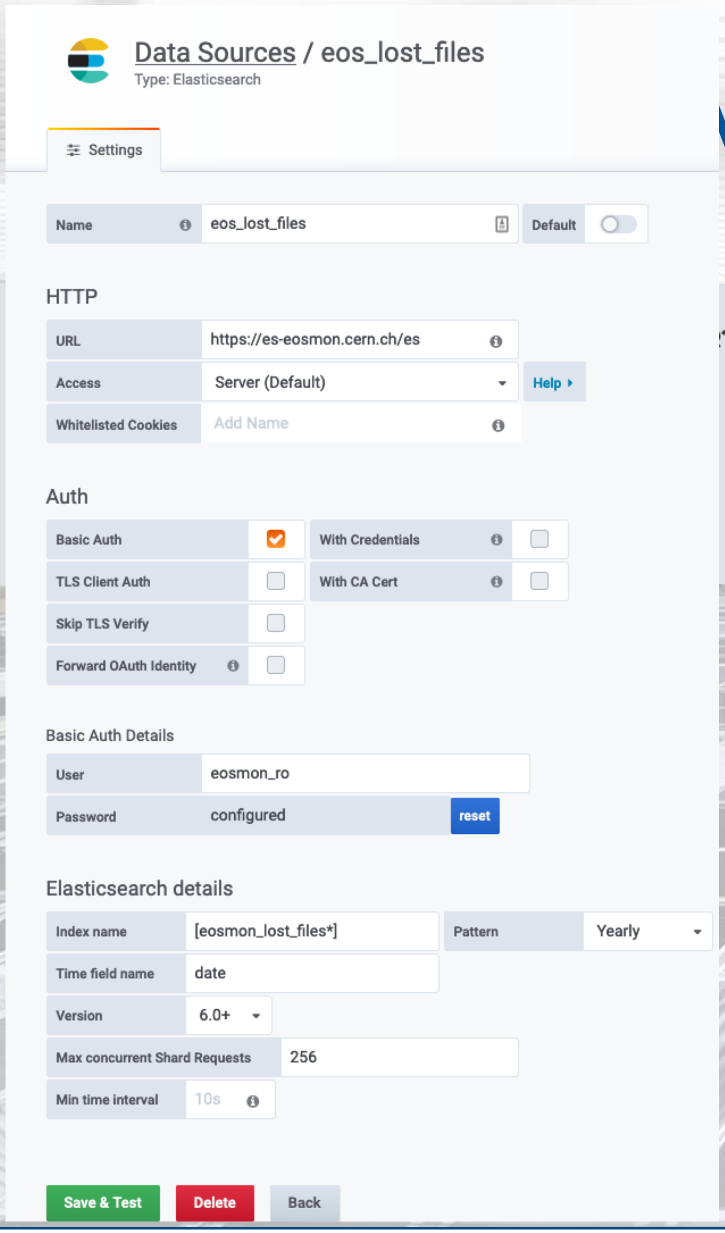
```
xstype: adler xs: 20902cff uid: 10367 size: 698,420 reason: no_replicas_found pxid: 000037f9 pid: 14329 nstripes: 2 nrep: 1 mtime: September 22nd 2017, 09:32:16.000 mode: 0664 mgm: eosalice-srv-m1.cern.ch lid: 00600112
layout: replica keylength.file: 61 ino: 300777143585996800 gid: 1395 fxid: 42c9337f fst.size: 698,420 fst.path: /eos/alice/grid/08/22095/2896bfcc-9f68-11e7-b9de-2b7740215b96 fst.nrep: 1 fst.fxid: 42c9337f fst.checksumtype: adle
r fst.checksum: 20902cff00000000000000000000000000000000 fsdel: 10427 file: /eos/alice/grid/08/22095/2896bfcc-9f68-11e7-b9de-2b7740215b96 fid: 1,120,482,175 experiment: alice etag: 300777143585996800:20902cff date: May 7th 2019, 0
2:00:00.000 ctime: September 22nd 2017, 09:32:16.000 clock: 0 _type: lost_files _score: 1 _index: eosmon_lost_files-2019 _id: 42c9337f

xstype: adler xs: 33243fa0 uid: 10367 size: 2,389 reason: no_replicas_found pxid: 0000091c pid: 2332 nstripes: 2 nrep: 1 mtime: March 6th 2015, 13:32:51.000 mode: 0664 mgm: eosalice-srv-m1.cern.ch lid: 00600112 layout: rep
lica keylength.file: 61 ino: 70278122783637504 gid: 1395 fxid: 0f9ad949 fst.size: 2,389 fst.path: /eos/alice/grid/08/22079/e53a5f0c-c3fc-11e4-b225-1774a3c593fa fst.nrep: 1 fst.fxid: 0f9ad949 fst.checksumtype: adler
fst.checksum: 33243fa000000000000000000000000000000000 fsdel: 7223 file: /eos/alice/grid/08/22079/e53a5f0c-c3fc-11e4-b225-1774a3c593fa fid: 261,806,409 experiment: alice etag: 70278122783637504:33243fa0 date: May 7th 2019, 02:00:0
0.000 ctime: March 6th 2015, 13:32:49.000 clock: 0 _type: lost_files _score: 1 _index: eosmon_lost_files-2019 _id: 0f9ad949
```

Metadata
(eos-ops-lostfiles)

kibana Data discovery

CERNBox

Missing files tracking



Data Sources / eos_lost_files
Type: Elasticsearch

Settings

Name	eos_lost_files	Default	off

HTTP

URL	https://es-eosmon.cern.ch/es
Access	Server (Default)
Whitelisted Cookies	Add Name

Auth

Basic Auth	✓	With Credentials	☐
TLS Client Auth	☐	With CA Cert	☐
Skip TLS Verify	☐		
Forward OAuth Identity	☐		

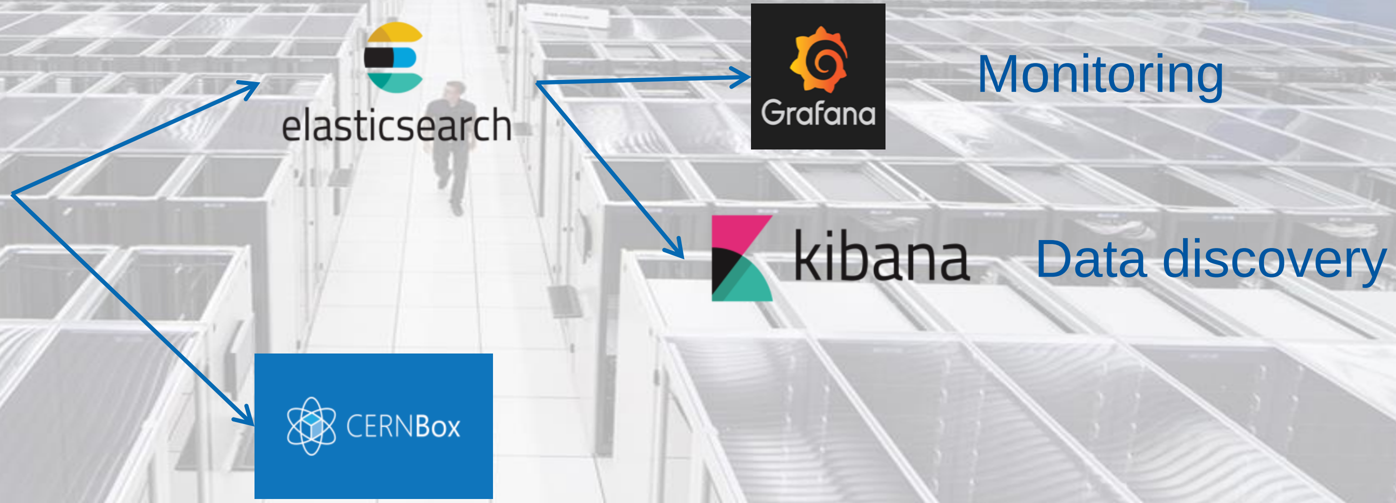
Basic Auth Details

User	eosmon_ro
Password	configured

Elasticsearch details

Index name	[eosmon_lost_files*]	Pattern	Yearly
Time field name	date		
Version	6.0+		
Max concurrent Shard Requests	256		
Min time interval	10s		

Save & Test | Delete | Back



Monitoring

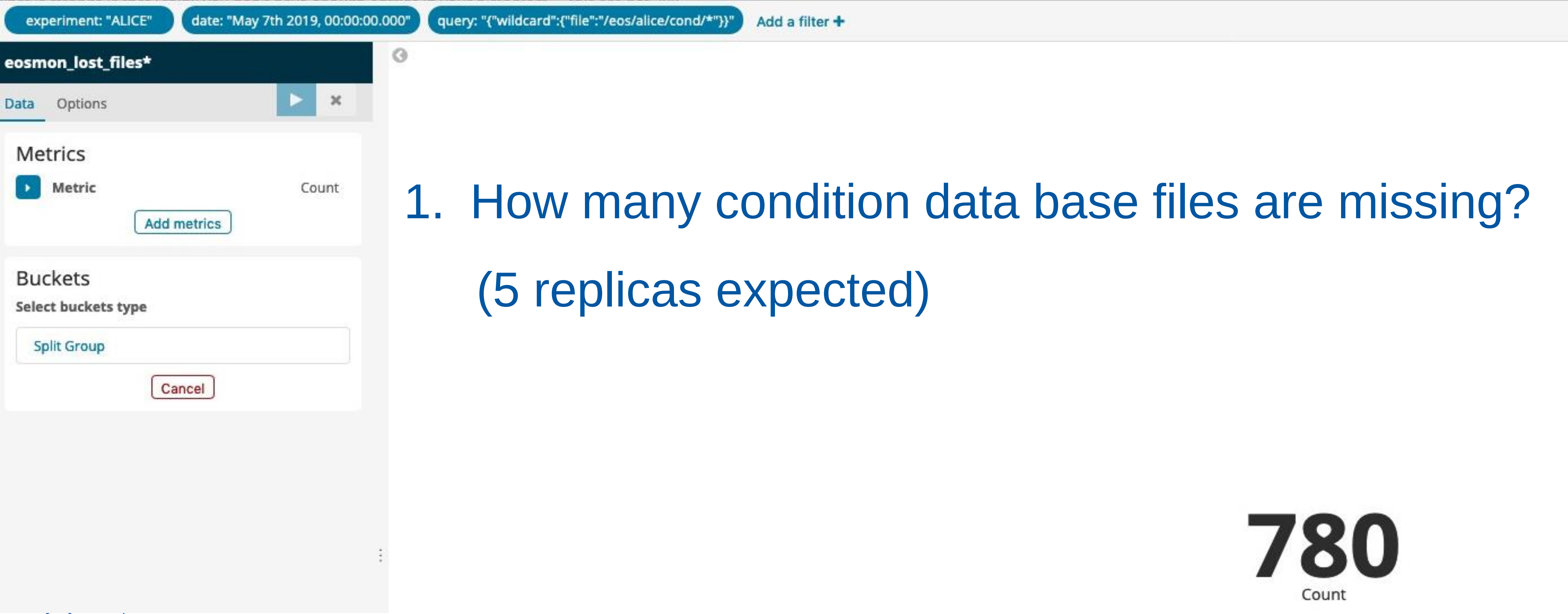
Data discovery

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



experiment: "ALICE" date: "May 7th 2019, 00:00:00.000" query: "{"wildcard":{"file":"/eos/alice/cond/*"}}"

eosmon_lost_files*

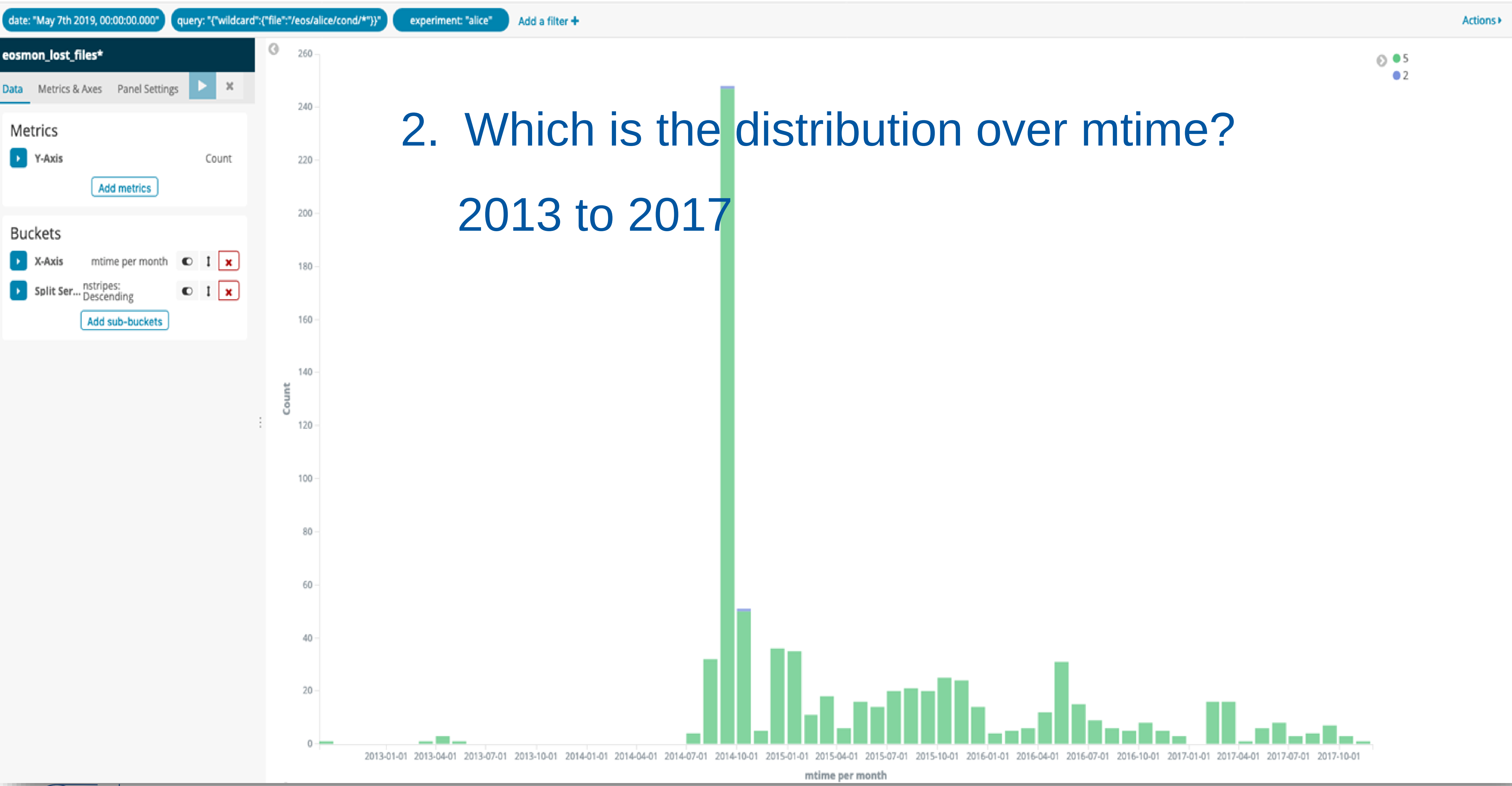
Metrics
Metric: Count

Buckets
Select buckets type: Split Group

780
Count

1. How many condition data base files are missing?

   (5 replicas expected)


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

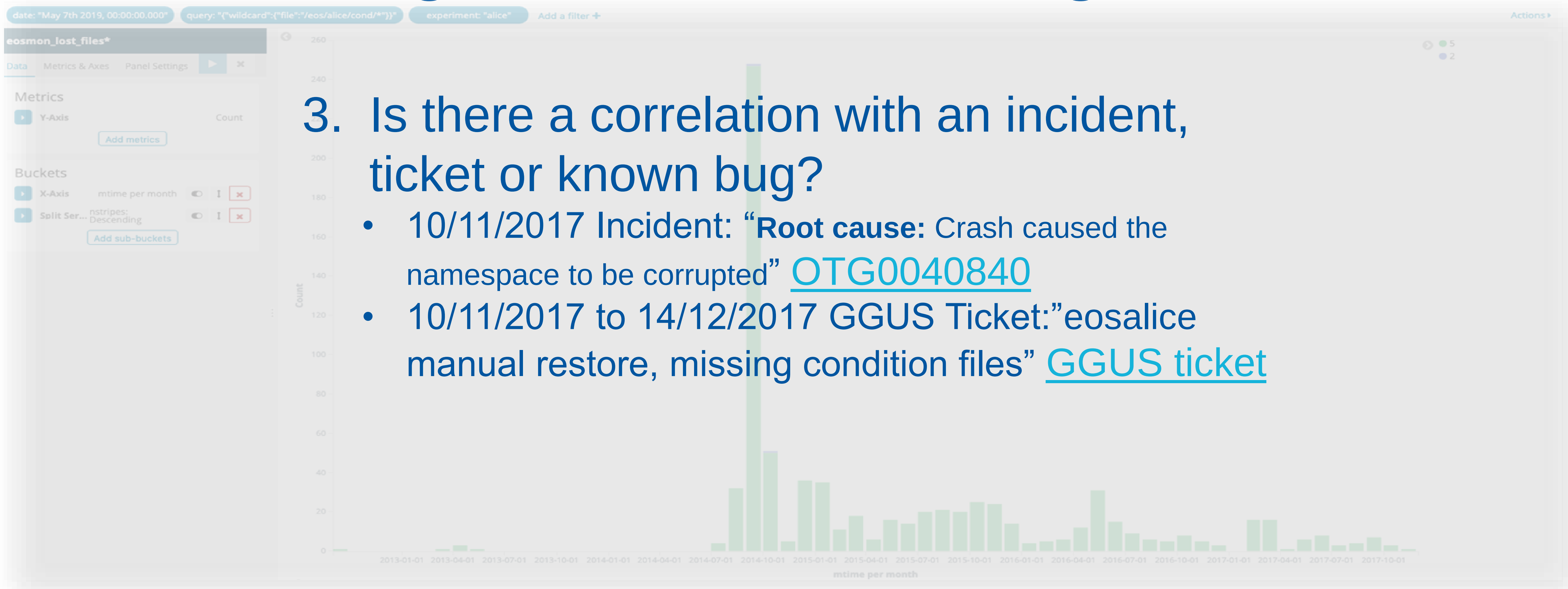
Metrics
Y-Axis: Count
Add metrics

Buckets
X-Axis: mtime per month
Split Ser...: nstripes: Descending
Add sub-buckets

2. Which is the distribution over mtime?

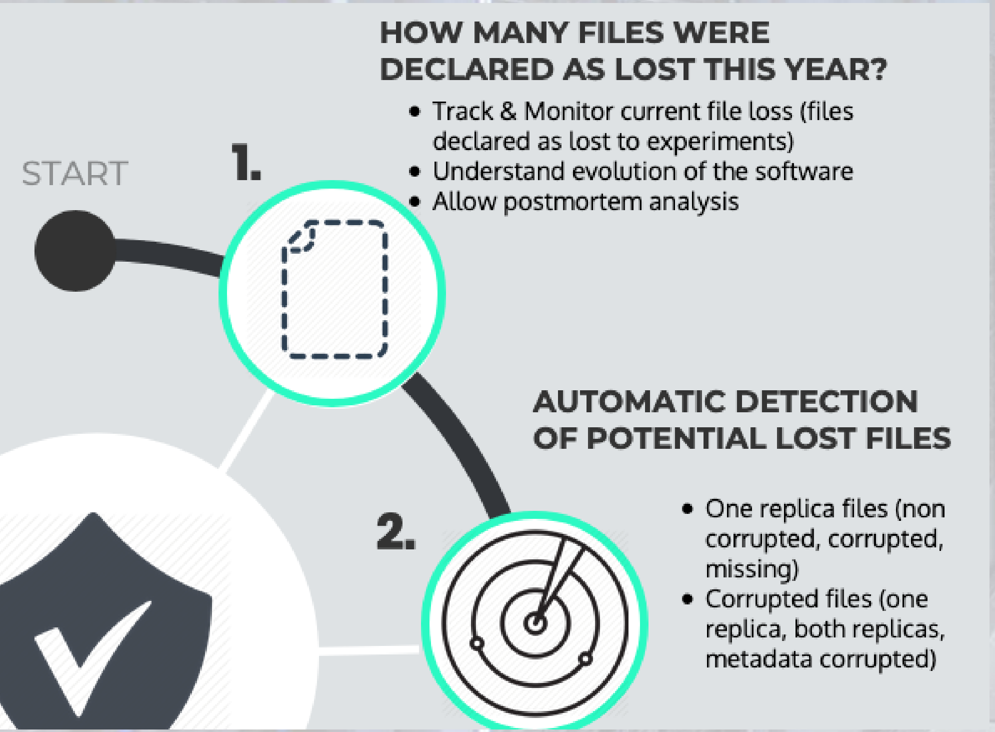
2013 to 2017

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


HOW MANY FILES WERE DECLARED AS LOST THIS YEAR?
START
1.
AUTOMATIC DETECTION OF POTENTIAL LOST FILES
2.

- Track & Monitor current file loss (files declared as lost to experiments)
- Understand evolution of the software
- Allow postmortem analysis

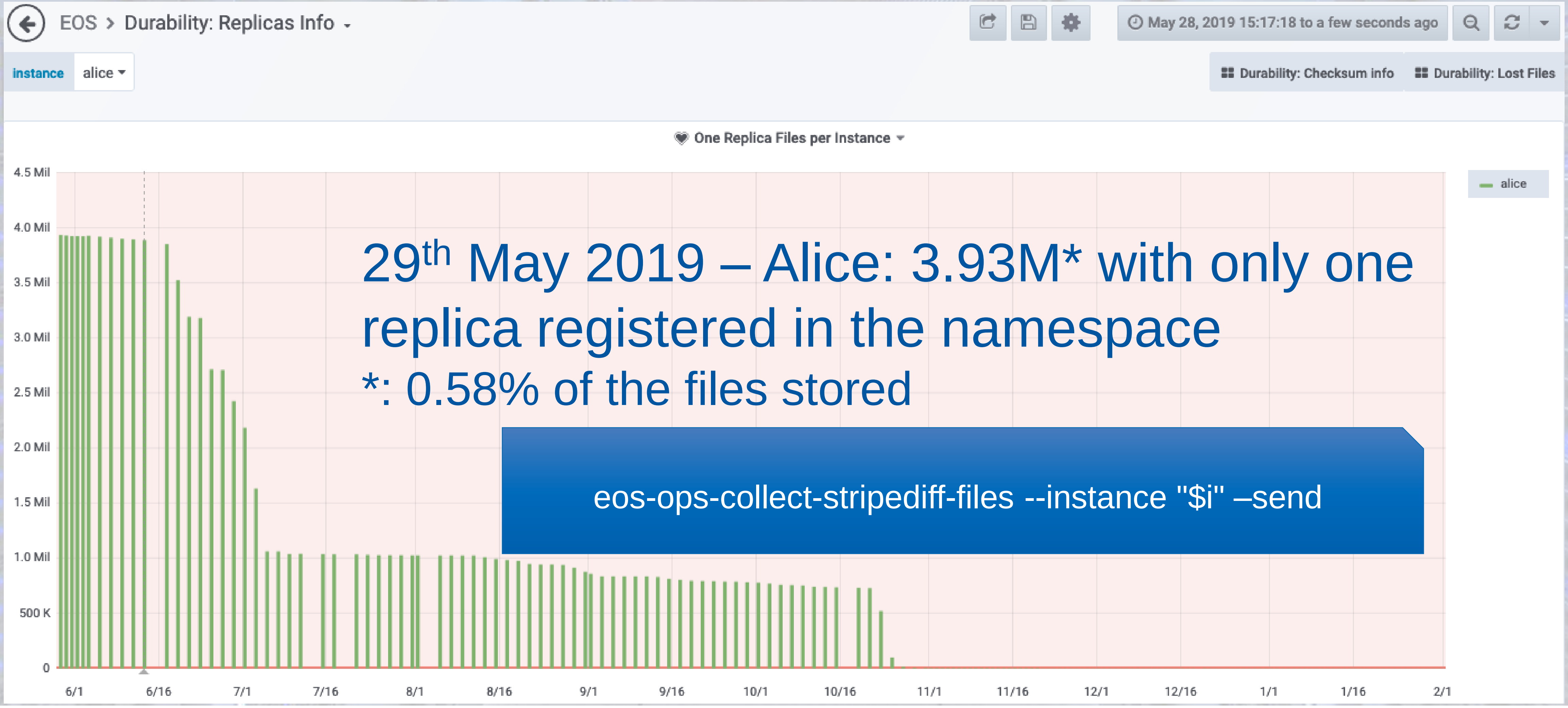
- One replica files (non corrupted, corrupted, missing)
- Corrupted files (one replica, both replicas, metadata corrupted)

- **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



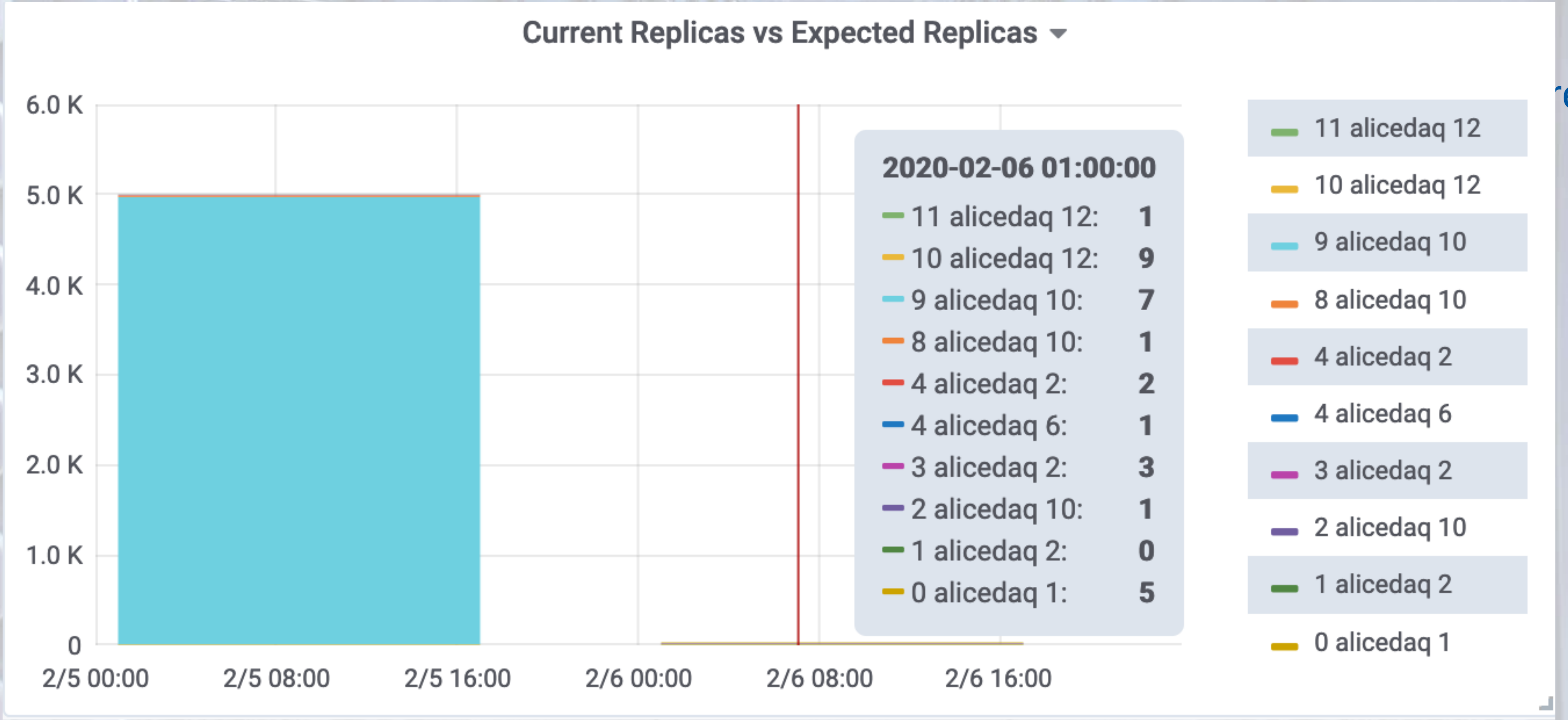
29th May 2019 – Alice: 3.93M* with only one replica registered in the namespace

*: 0.58% of the files stored

```
eos-ops-collect-stripediff-files --instance "$i" –send
```

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

Under-replication of Rain files in Alicedaq



2020-02-06 01:00:00

11 alicedaq 12:	1
10 alicedaq 12:	9
9 alicedaq 10:	7
8 alicedaq 10:	1
4 alicedaq 2:	2
4 alicedaq 6:	1
3 alicedaq 2:	3
2 alicedaq 10:	1
1 alicedaq 2:	0
0 alicedaq 1:	5

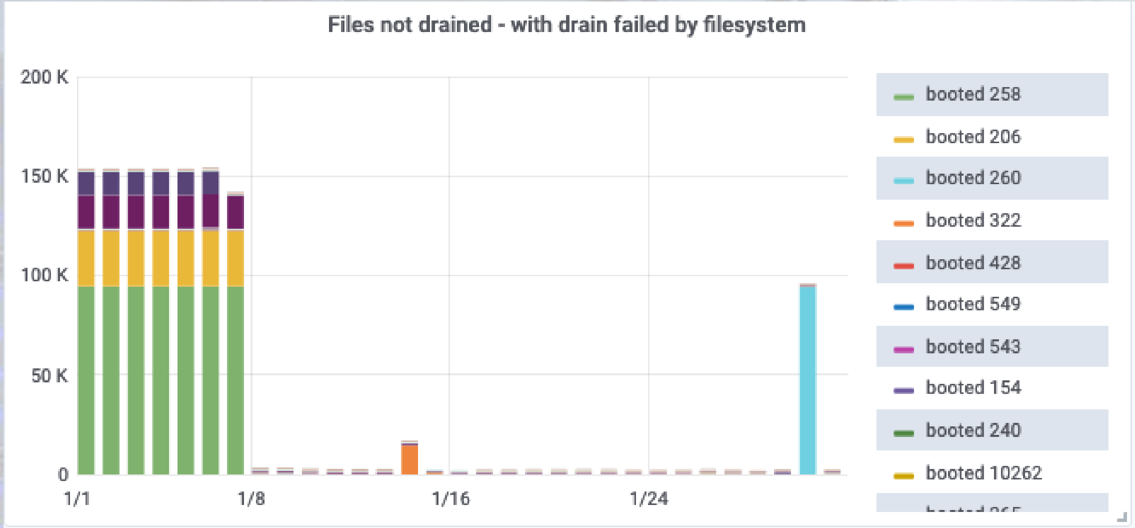
- 11 alicedaq 12
- 10 alicedaq 12
- 9 alicedaq 10
- 8 alicedaq 10
- 4 alicedaq 2
- 4 alicedaq 6
- 3 alicedaq 2
- 2 alicedaq 10
- 1 alicedaq 2
- 0 alicedaq 1

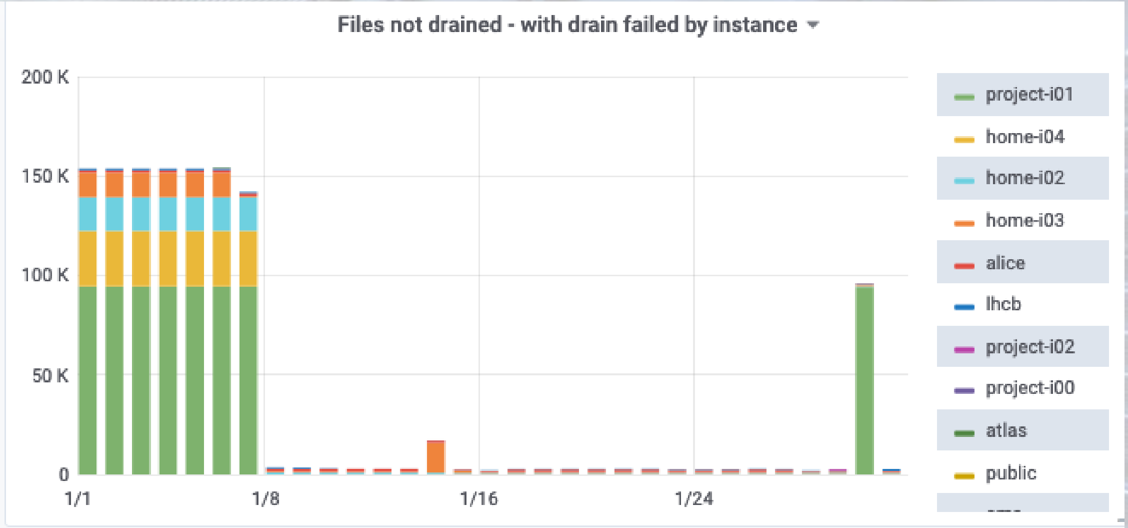
red

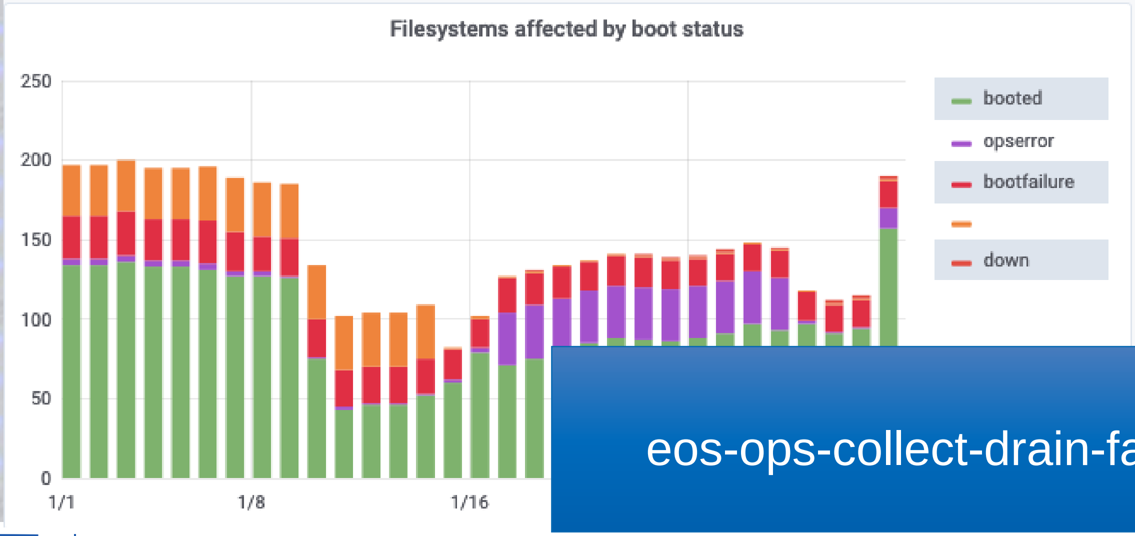
```
eos-ops-collect-stripediff-files --instance "$i" –send
```

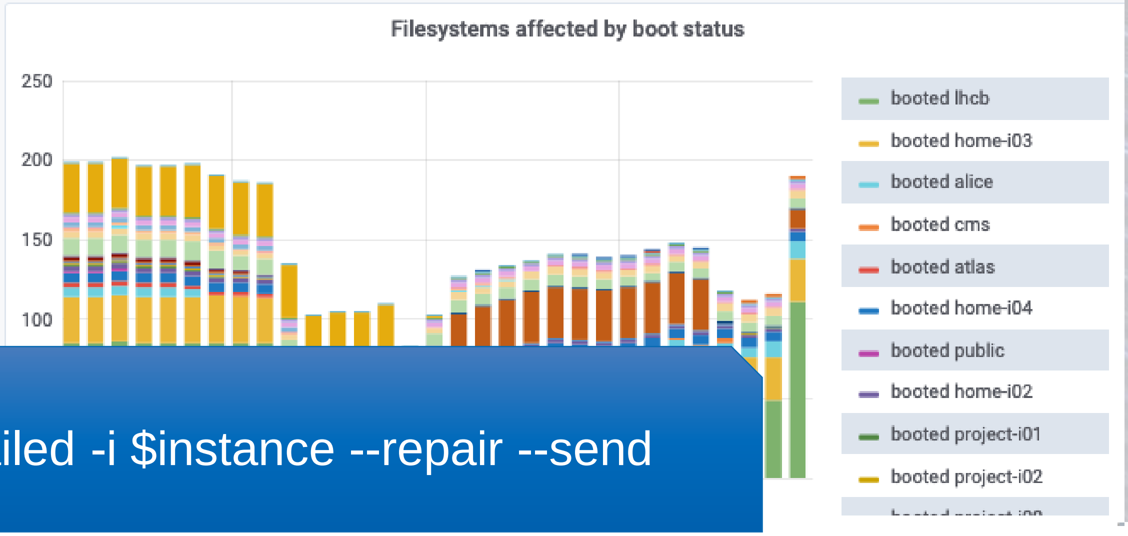
Automatic detection of ***high risk*** files

Draining failures – last month (January 2020)



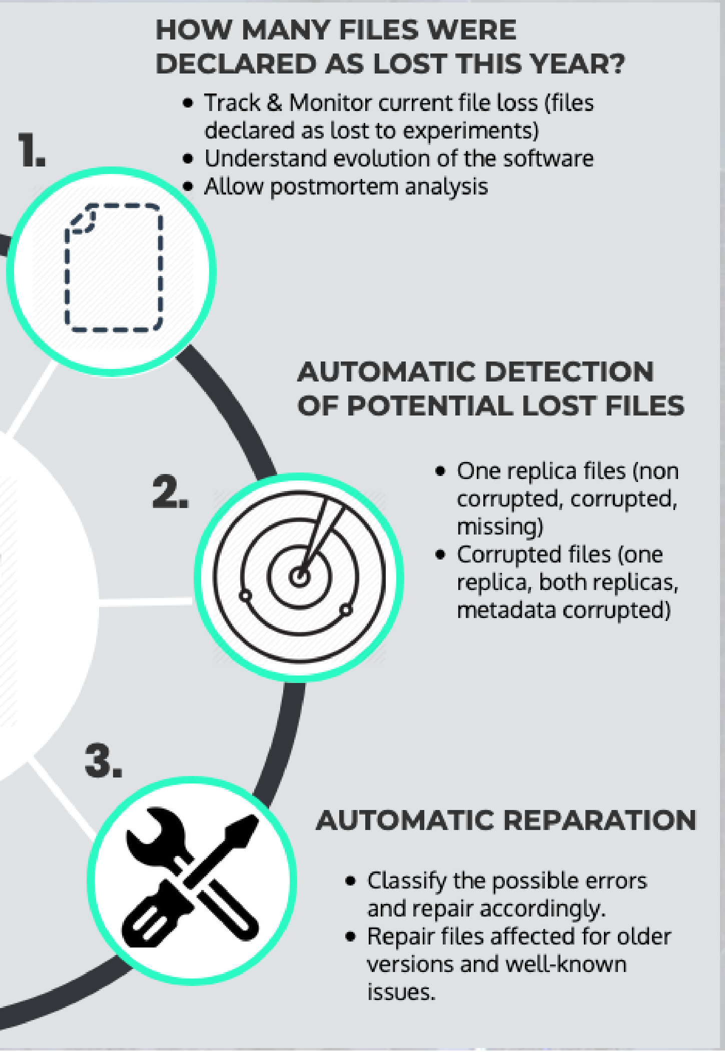






```
eos-ops-collect-drain-failed -i $instance --repair --send
```

Automatic reparation



1\.

HOW MANY FILES WERE DECLARED AS LOST THIS YEAR?

- Track & Monitor current file loss (files declared as lost to experiments)
- Understand evolution of the software
- Allow postmortem analysis

2\.

AUTOMATIC DETECTION OF POTENTIAL LOST FILES

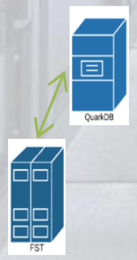
- One replica files (non corrupted, corrupted, missing)
- Corrupted files (one replica, both replicas, metadata corrupted)

3\.

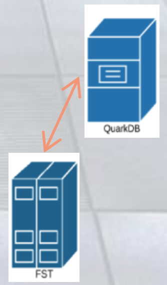
AUTOMATIC REPARATION

- Classify the possible errors and repair accordingly.
- Repair files affected for older versions and well-known issues.

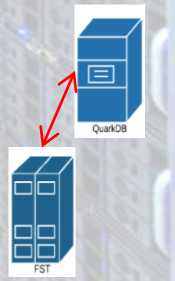
- Namespace full scan is not enough
  - We need to go deeper and get the storage nodes information
- Classify the possible errors and repair accordingly
  - **Divide & Conquer:** 14 categories for one replica files out of 21 categories in total
- Repair files affected for older versions and well-known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



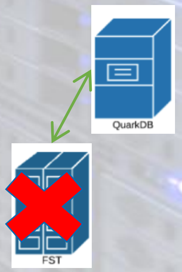
One replica and size and checksum match the namespace



One replica with checksum 0 in namespace

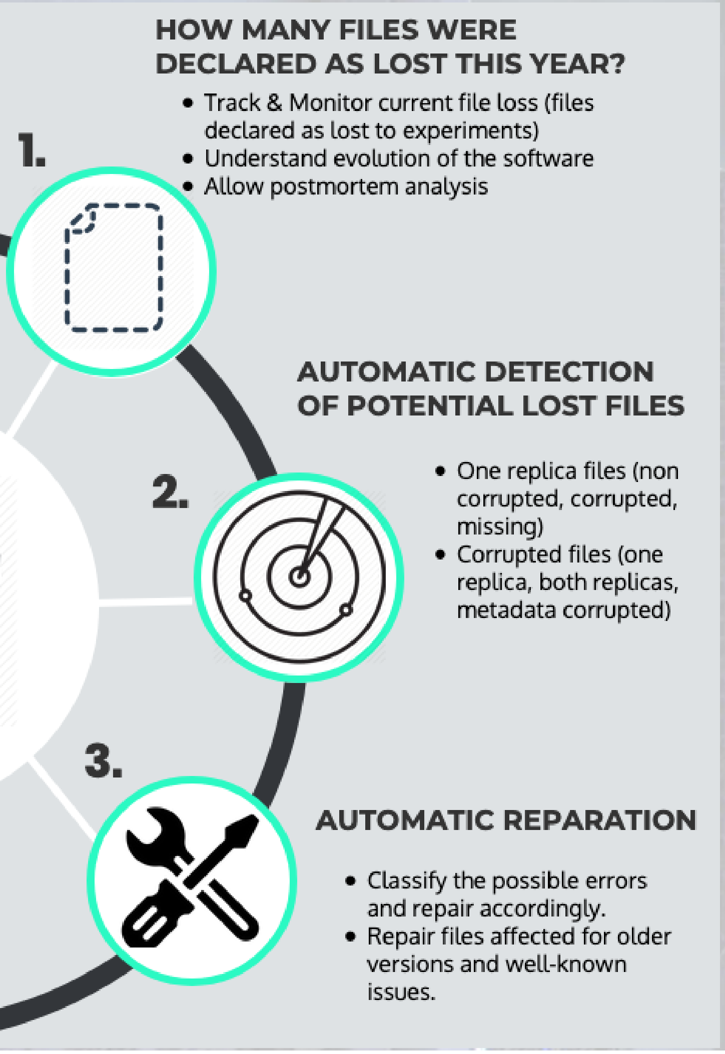


One replica with size and checksum mismatch with namespace



Missing replica

Automatic reparation



HOW MANY FILES WERE DECLARED AS LOST THIS YEAR?

1.
- Track & Monitor current file loss (files declared as lost to experiments)
- Understand evolution of the software
- Allow postmortem analysis

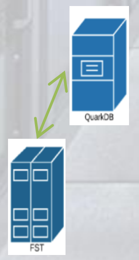
AUTOMATIC DETECTION OF POTENTIAL LOST FILES

2.
- One replica files (non corrupted, corrupted, missing)
- Corrupted files (one replica, both replicas, metadata corrupted)

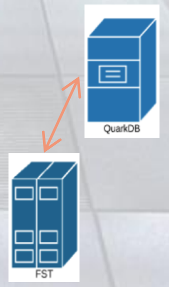
AUTOMATIC REPARATION

3.
- Classify the possible errors and repair accordingly.
- Repair files affected for older versions and well-known issues.

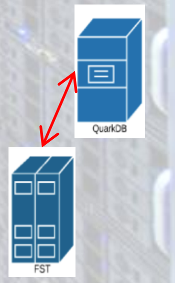
- Namespace full scan is not enough
  - We need to go deeper and get the storage nodes information
- Classify the possible errors and repair accordingly
  - **Divide & Conquer:** 14 categories for one replica files out of 21 categories in total
- Repair files affected for older versions and well-known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



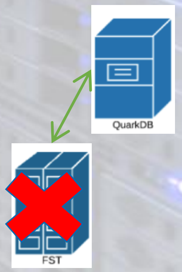
One replica and size and checksum match the namespace



One replica with checksum 0 in namespace



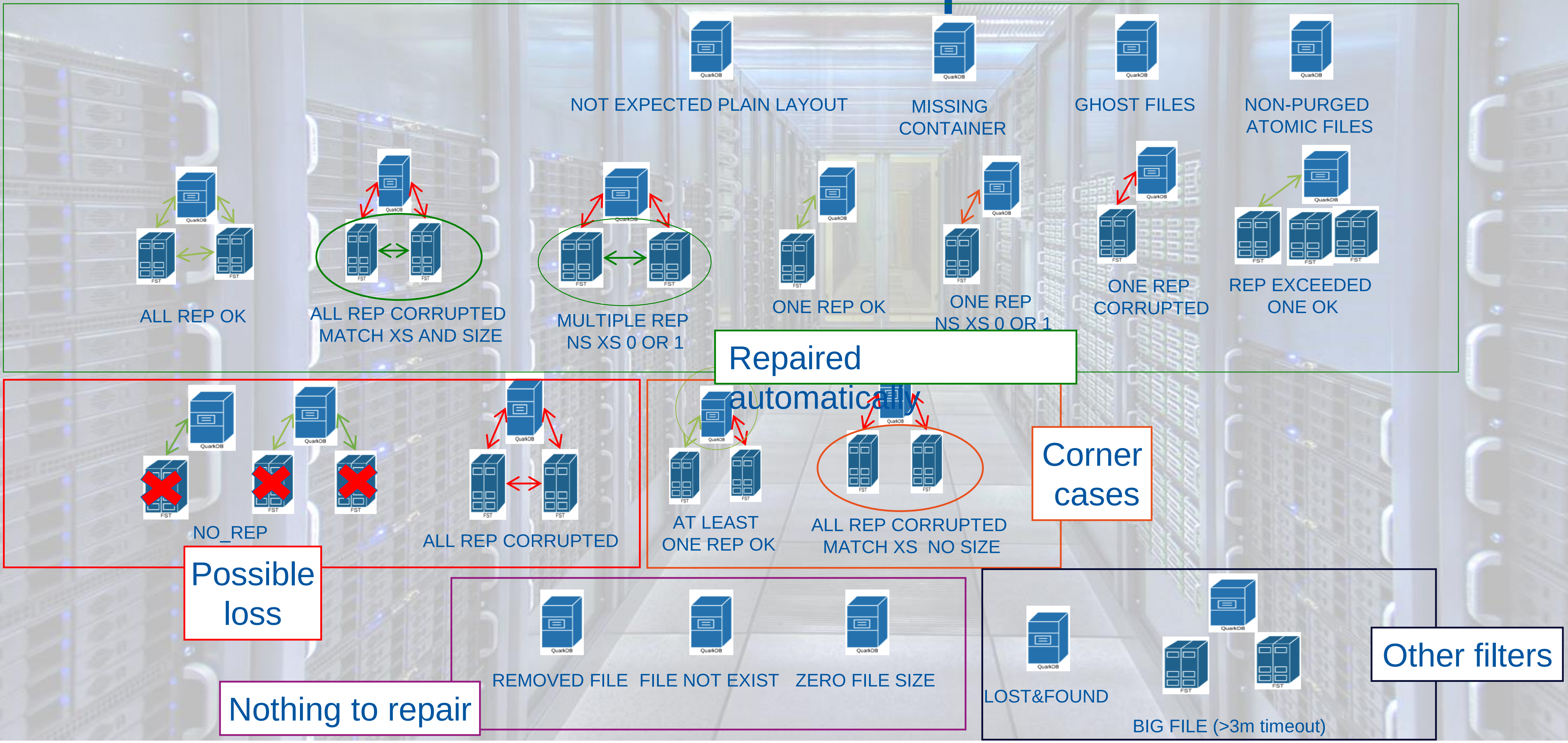
One replica with size and checksum mismatch with namespace



Missing replica

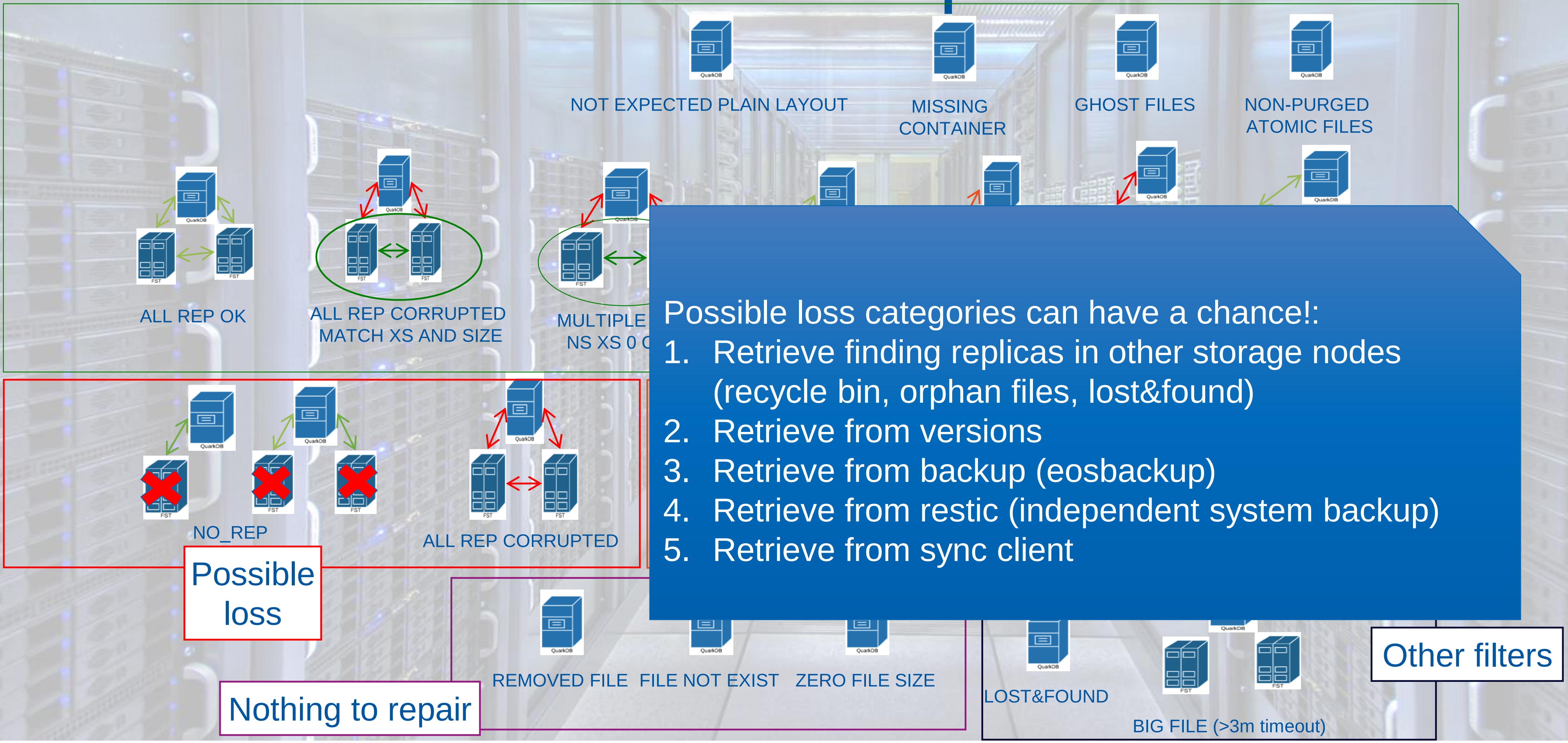
Automatic reparation

21 + 1 categories



Automatic reparation

21 + 1 categories

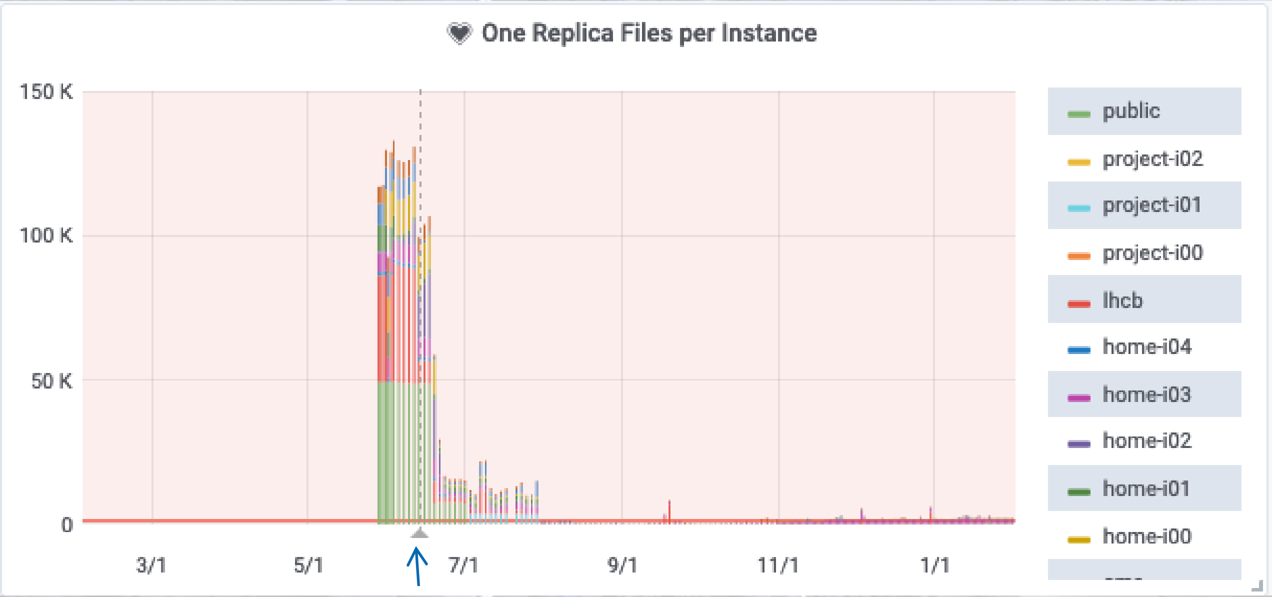


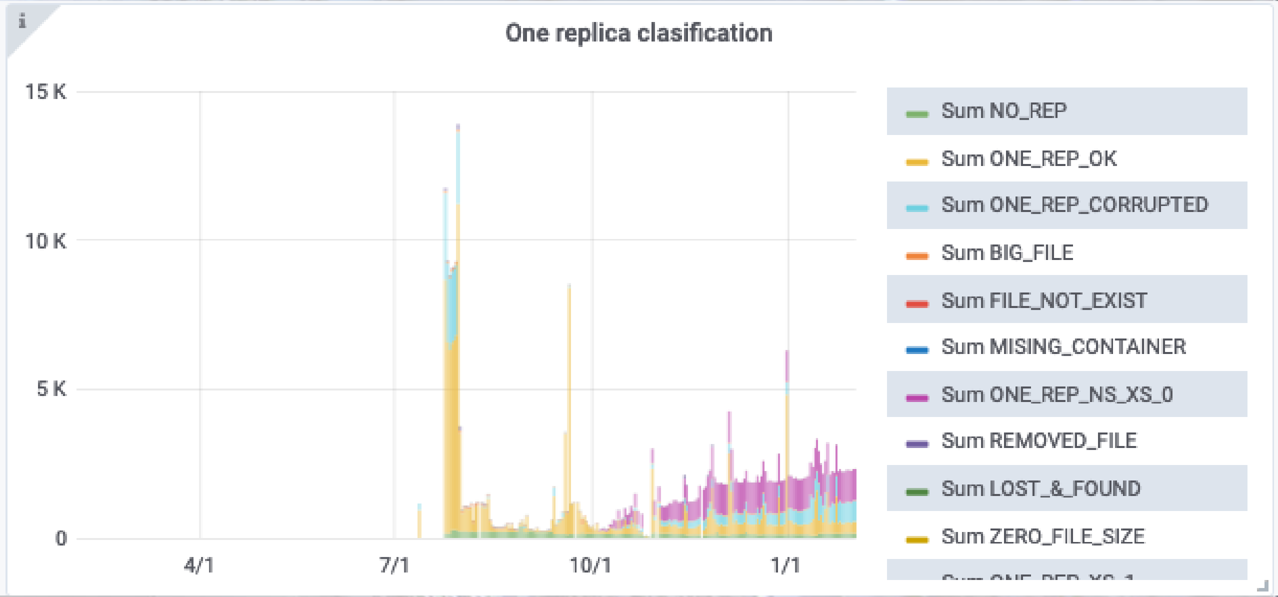
Possible loss categories can have a chance!:

1. Retrieve finding replicas in other storage nodes (recycle bin, orphan files, lost&found)
2. Retrieve from versions
3. Retrieve from backup (eosbackup)
4. Retrieve from restic (independent system backup)
5. Retrieve from sync client

Automatic reparation

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

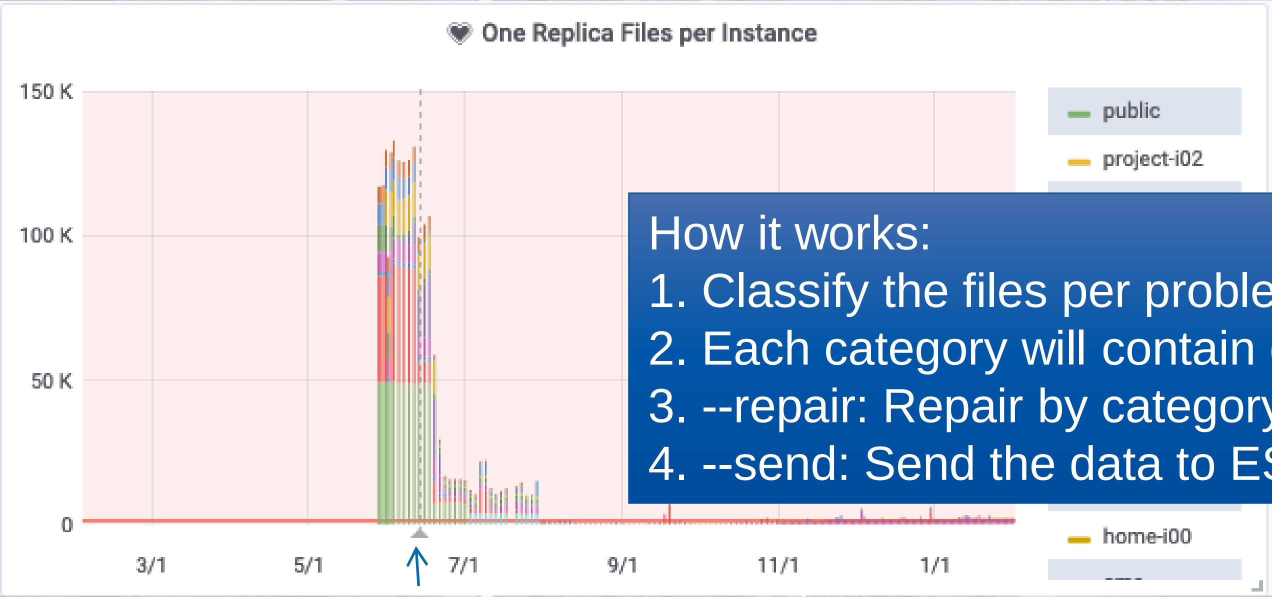


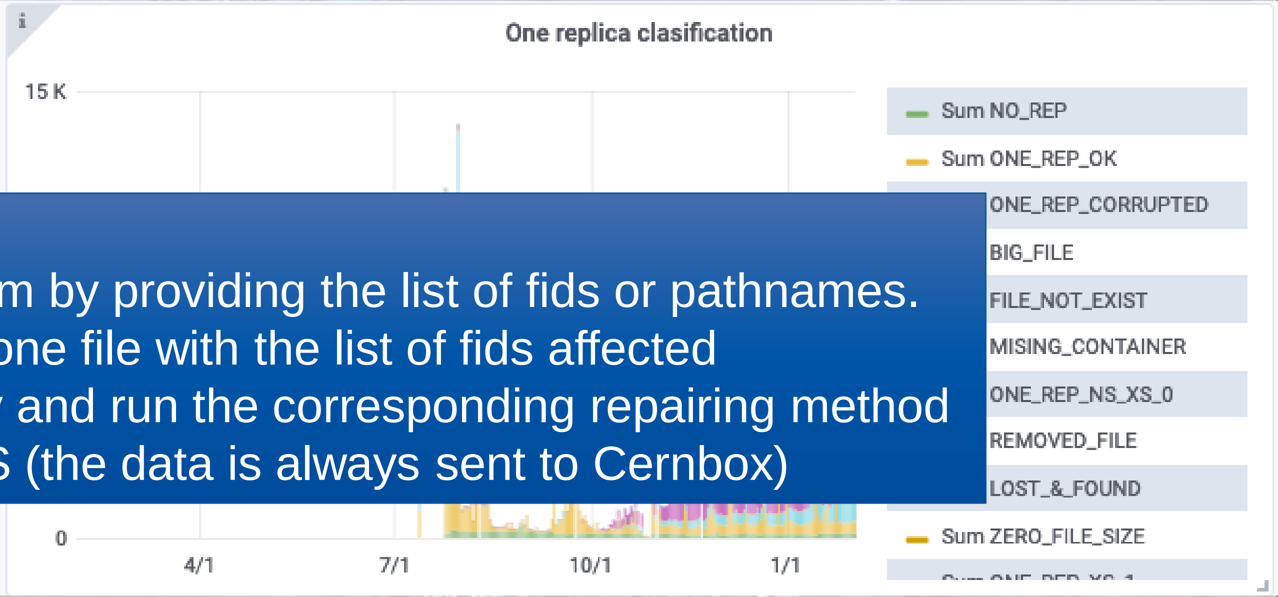


```
eos-ops-repair -f "$i"_one_replica_$(date +%Y-%m-%d -d "yesterday").txt --id_type dec -i "$i" -l DEBUG --rundeck
--one_rep --repair --send
```

Automatic reparation

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





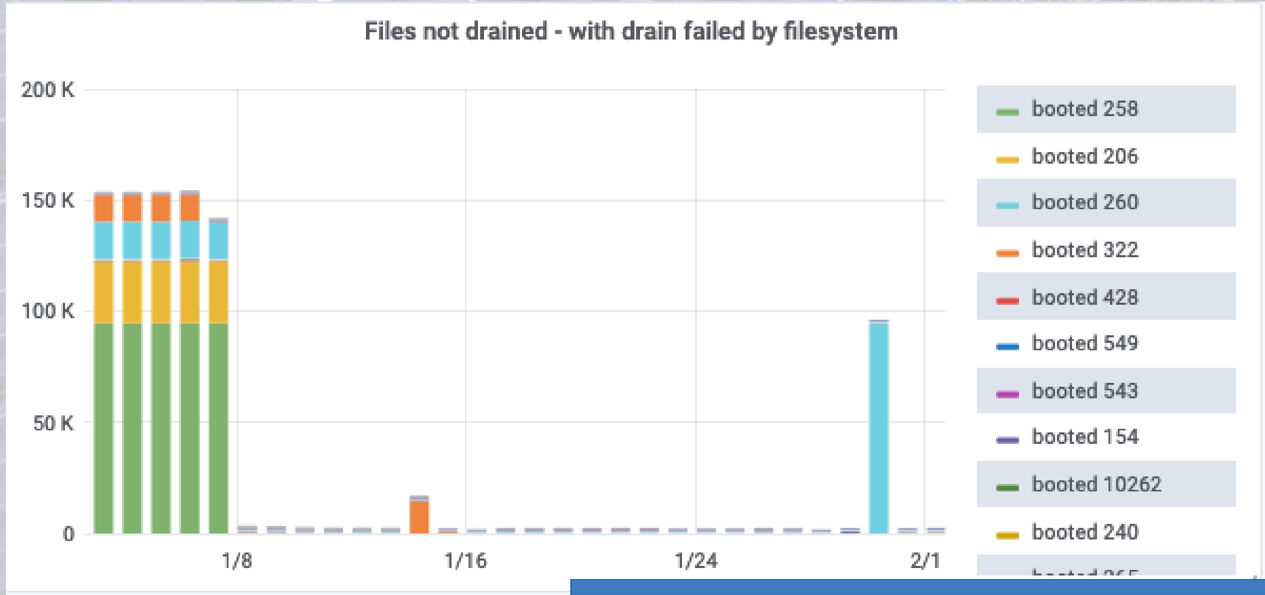
How it works:

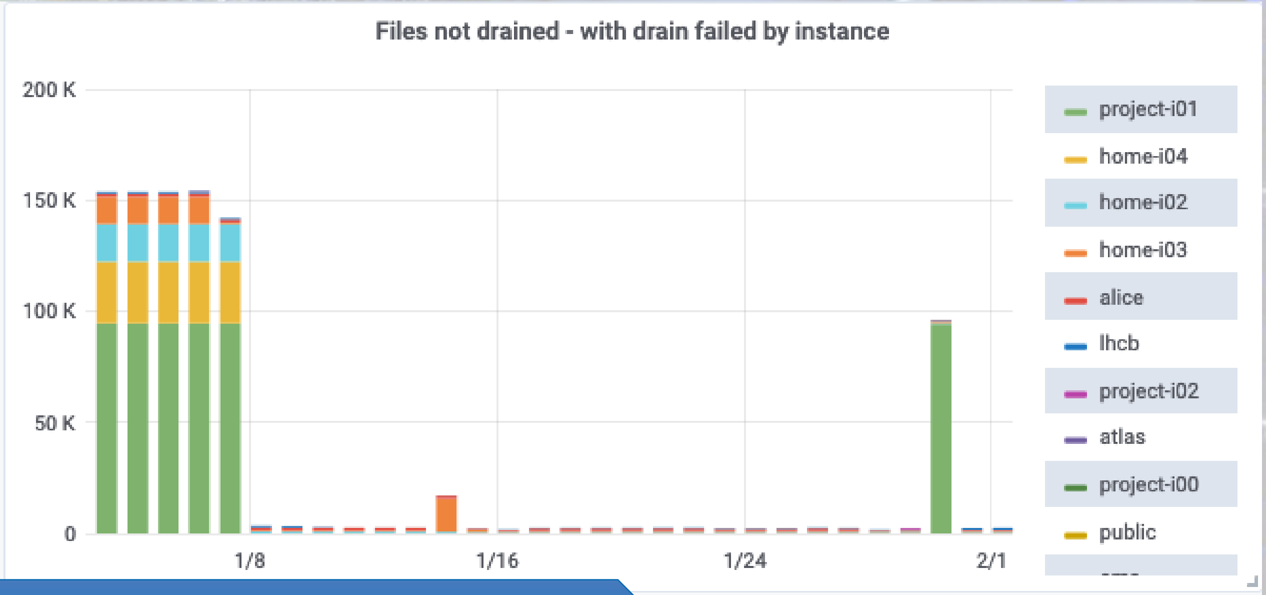
1. Classify the files per problem by providing the list of fids or pathnames.
2. Each category will contain one file with the list of fids affected
3. --repair: Repair by category and run the corresponding repairing method
4. --send: Send the data to ES (the data is always sent to Cernbox)

```
eos-ops-repair -f "$i"_one_replica_$(date +%Y-%m-%d -d "yesterday").txt --id_type dec -i "$i" -l DEBUG --rundeck
--one_rep --repair --send
```

Automatic reparation

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



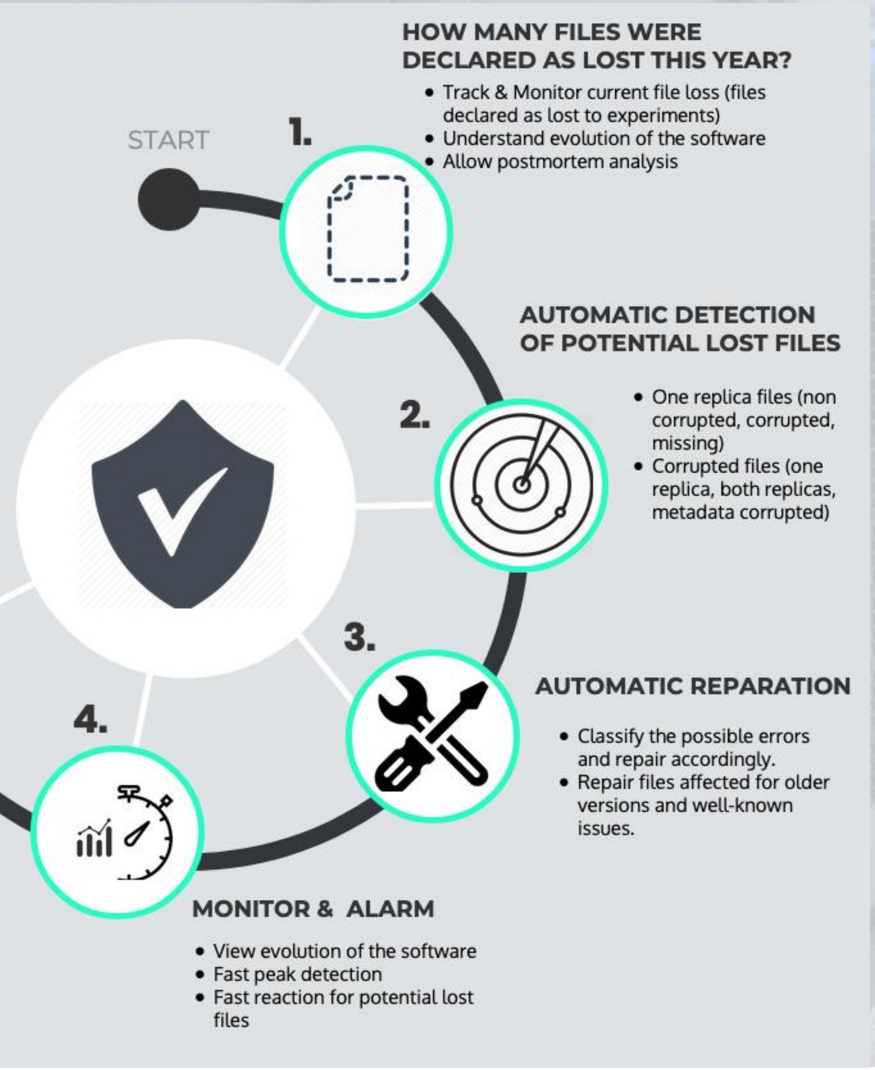


```
eos-ops-collect-drain-failed -i $instance --repair --send
```

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



START

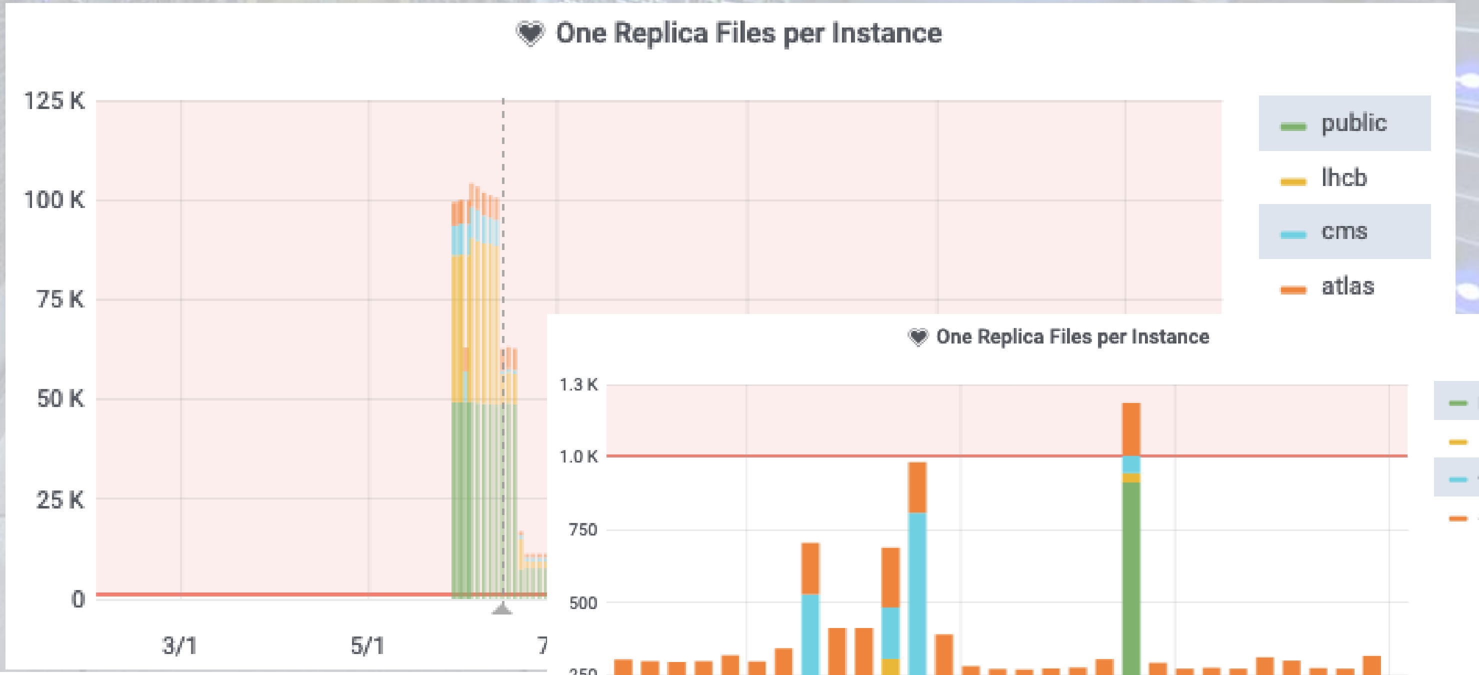
1. **HOW MANY FILES WERE DECLARED AS LOST THIS YEAR?**
- Track & Monitor current file loss (files declared as lost to experiments)
- Understand evolution of the software
- Allow postmortem analysis

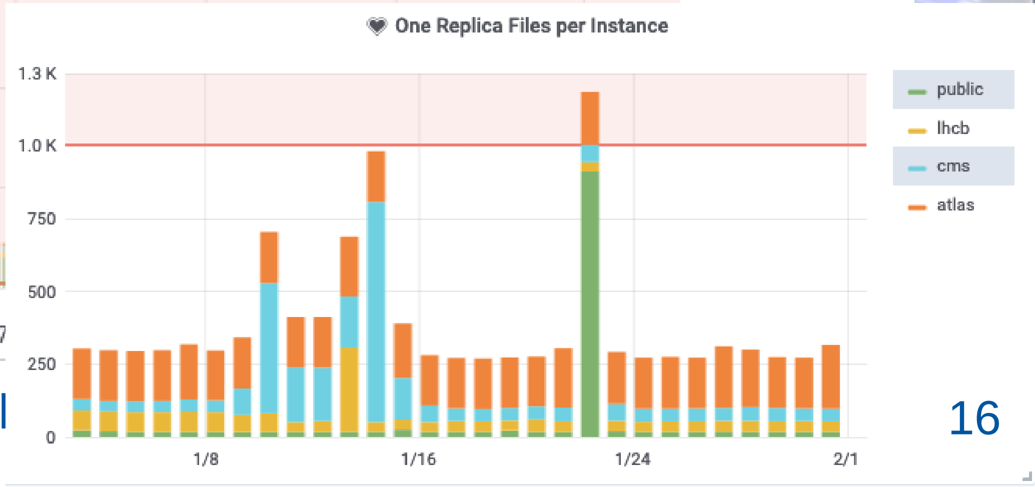
2. **AUTOMATIC DETECTION OF POTENTIAL LOST FILES**
- One replica files (non corrupted, corrupted, missing)
- Corrupted files (one replica, both replicas, metadata corrupted)

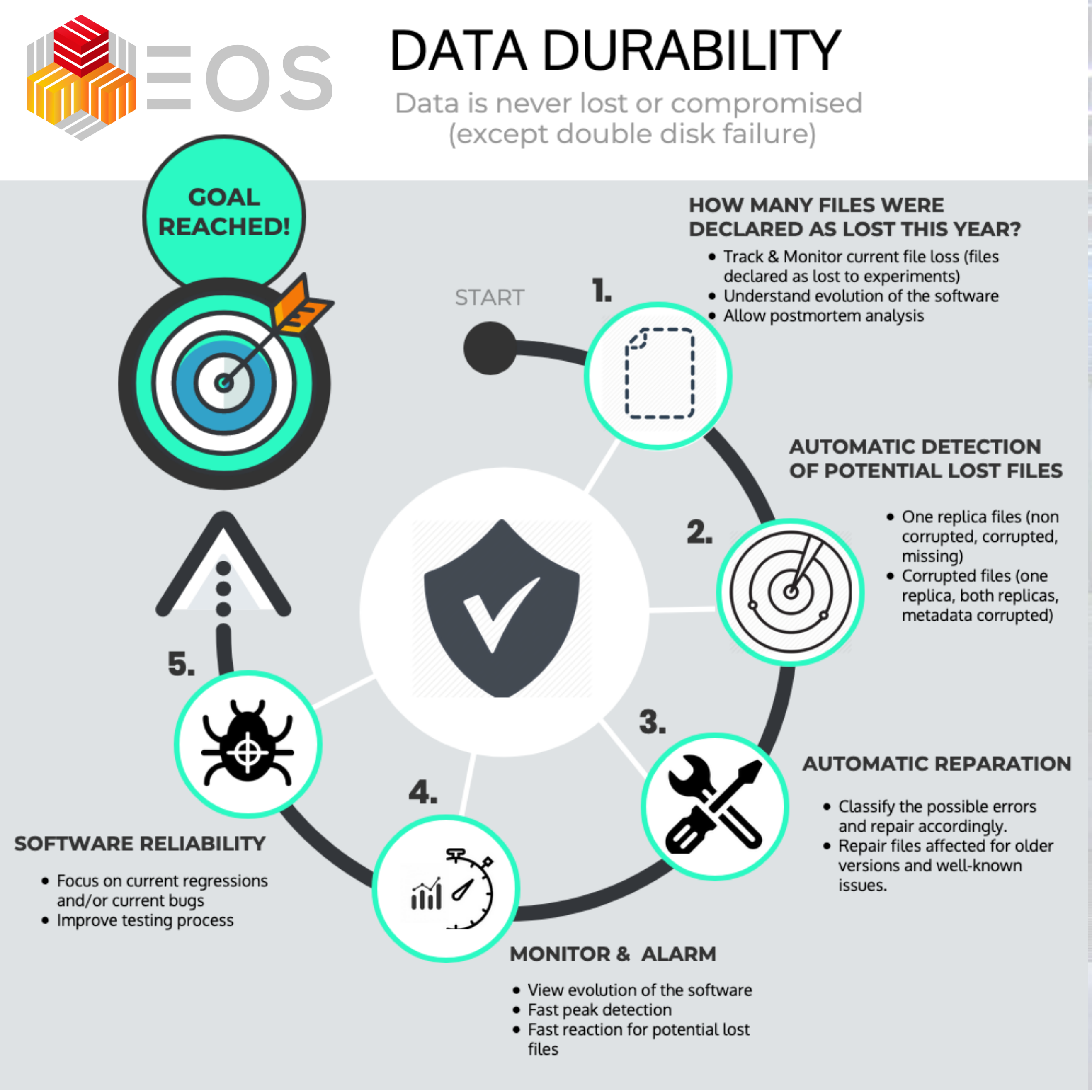
3. **AUTOMATIC REPARATION**
- Classify the possible errors and repair accordingly.
- Repair files affected for older versions and well-known issues.

4. **MONITOR & ALARM**
- View evolution of the software
- Fast peak detection
- Fast reaction for potential lost files

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






EOS
DATA DURABILITY
Data is never lost or compromised
(except double disk failure)
GOAL REACHED!
START
1.
HOW MANY FILES WERE DECLARED AS LOST THIS YEAR?
• Track & Monitor current file loss (files declared as lost to experiments)
• Understand evolution of the software
• Allow postmortem analysis
2.
AUTOMATIC DETECTION OF POTENTIAL LOST FILES
• One replica files (non corrupted, corrupted, missing)
• Corrupted files (one replica, both replicas, metadata corrupted)
3.
AUTOMATIC REPARATION
• Classify the possible errors and repair accordingly.
• Repair files affected for older versions and well-known issues.
4.
MONITOR & ALARM
• View evolution of the software
• Fast peak detection
• Fast reaction for potential lost files
5.
SOFTWARE RELIABILITY
• Focus on current regressions and/or current bugs
• Improve testing process

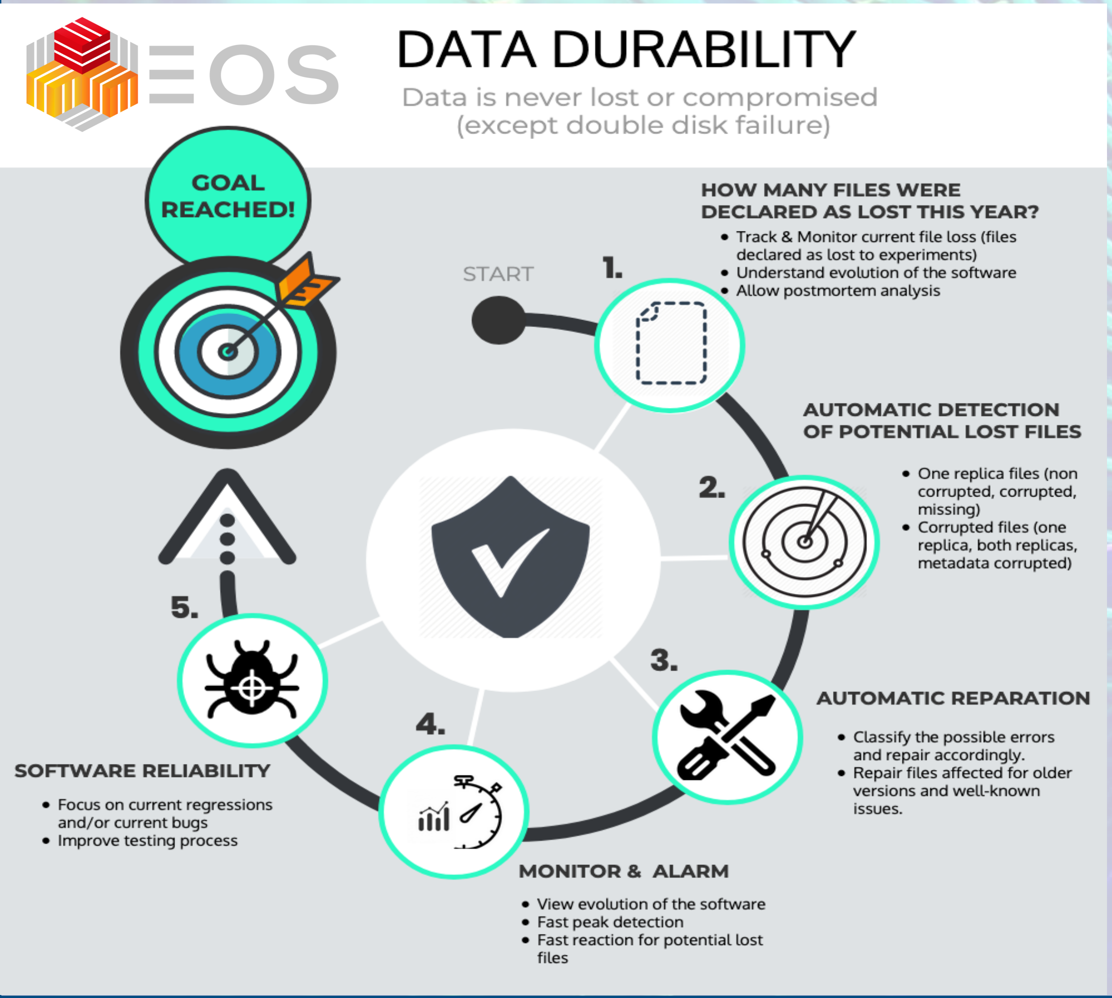
Software reliability

- 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



DATA DURABILITY

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

GOAL REACHED!

START

1. HOW MANY FILES WERE DECLARED AS LOST THIS YEAR?
- Track & Monitor current file loss (files declared as lost to experiments)
- Understand evolution of the software
- Allow postmortem analysis

2. AUTOMATIC DETECTION OF POTENTIAL LOST FILES
- One replica files (non corrupted, corrupted, missing)
- Corrupted files (one replica, both replicas, metadata corrupted)

3. AUTOMATIC REPARATION
- Classify the possible errors and repair accordingly.
- Repair files affected for older versions and well-known issues.

4. MONITOR & ALARM
- View evolution of the software
- Fast peak detection
- Fast reaction for potential lost files

5. SOFTWARE RELIABILITY
- Focus on current regressions and/or current bugs
- Improve testing process

- 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