Collection and harmonization of system logs and prototypal Analytics services with the Elastic (ELK) suite at the INFN-CNAF computing centre

Tommaso Diotalevi
(INFN and University of Bologna)

Co-authors:

Diego Michelotto (INFN-CNAF)
Lucia Morganti (INFN-CNAF)
Antonio Falabella (INFN-CNAF)
Barbara Martelli (INFN-CNAF)

Luca Giommi (INFN and University of Bologna)
Daniele Bonacorsi (University of Bologna)
Simone Rossi Tisbeni (University of Bologna)

INFN CNAF

INFN

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Taipei - April 3, 2019

International Symposium on Grids & Clouds 2019

Outline

- ✓ Collection of StoRM logs coming from different machines @INFN-CNAF Tier-1.
- ✓ Extraction of the information coming from such logs, using the ELK Stack suite.
- ✓ Creation of new visualisations and dashboards.
- ✓ Application of Machine learning algorithms for a preliminary predictive maintenance proof of concept.

The INFN-CNAF Computing Center

CNAF is the Italian national data center of INFN (Italian Institute for Nuclear Physics).

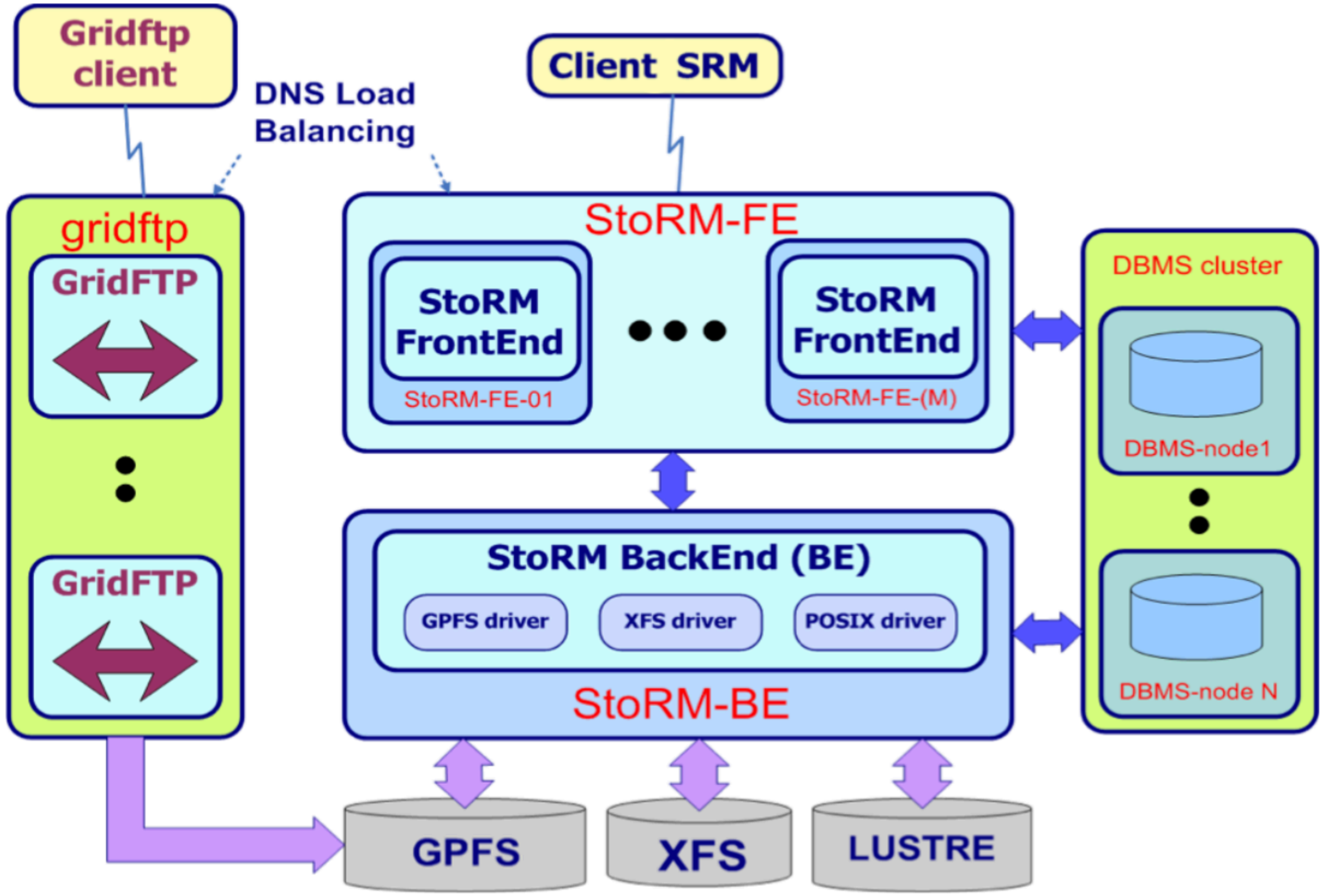
Since 2003, CNAF has hosted the Italian **Tier-1** for the High-Energy Physics experiments at the LHC in Geneva, as well as many other non-LHC experiments, as part of the WLCG.

StoRM

High performance disk-storage solutions are becoming increasingly important to deal with **large I/O throughput** required by HEP community.

Development and implementation of an ***SRM*** interface.

StoRM is a SRM service that relies on a parallel f.s. like GPFS.



StoRM

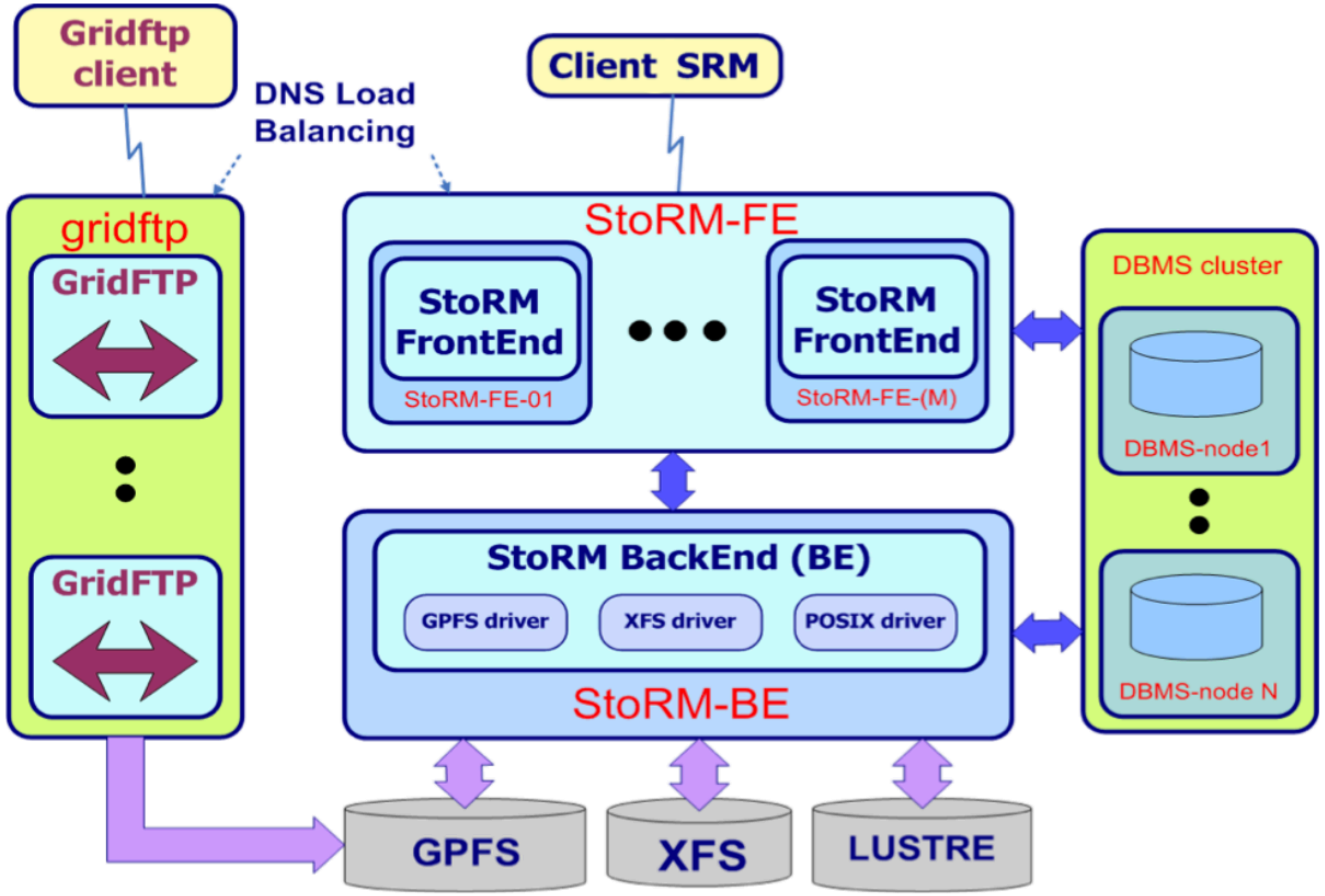
Two main components: *frontend* and *backend.*

- web service interface
- manage user authentication
- store request data in DB

Multiple instances on different nodes

- execute SRM operations
- management of files and space
- authorisation permissions and interaction with Grid services.

One shared instance



StoRM

Two main components: *frontend* and *backend.*

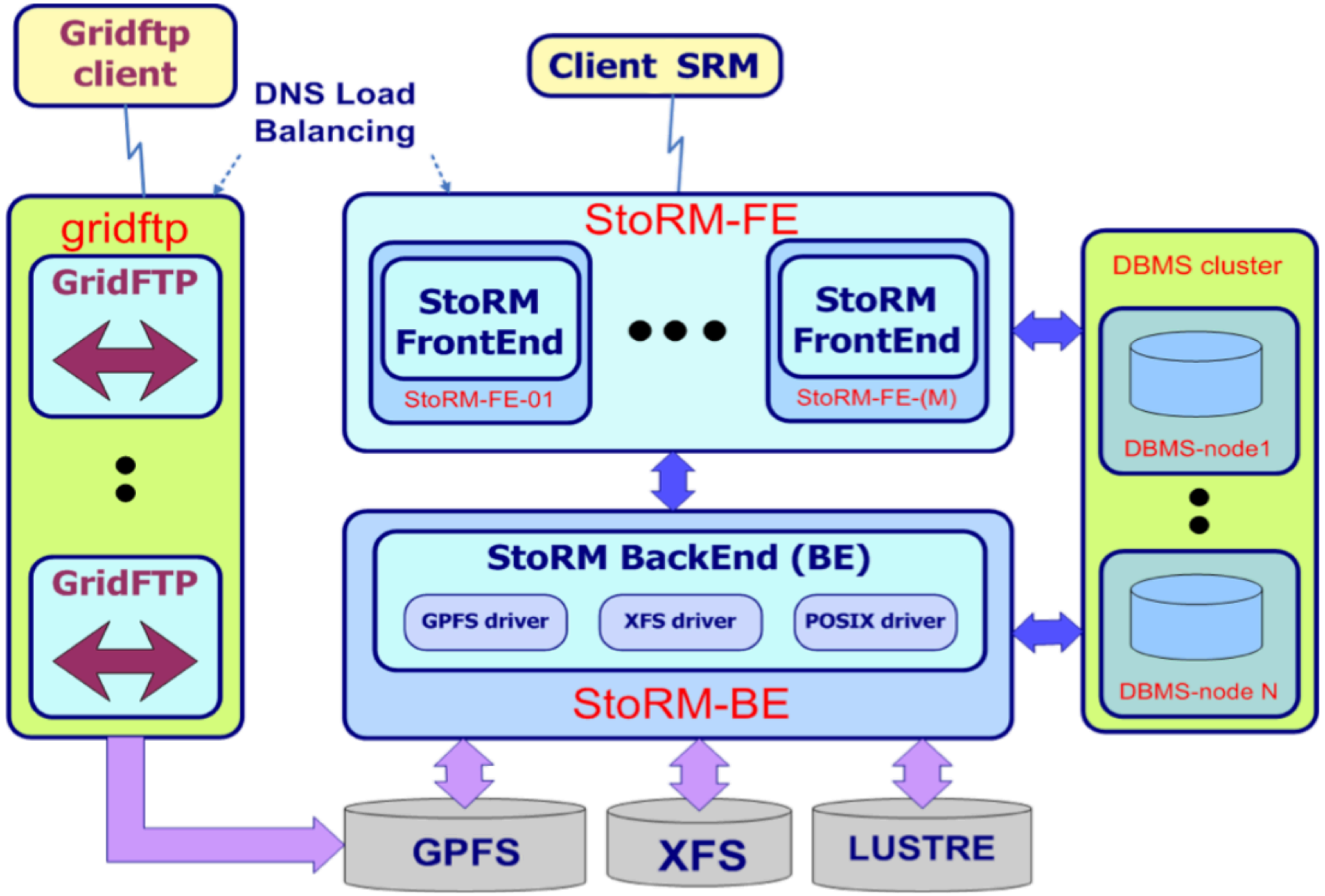
- web service interface
- manage user authentication
- store request data in DB

Multiple instances on different nodes

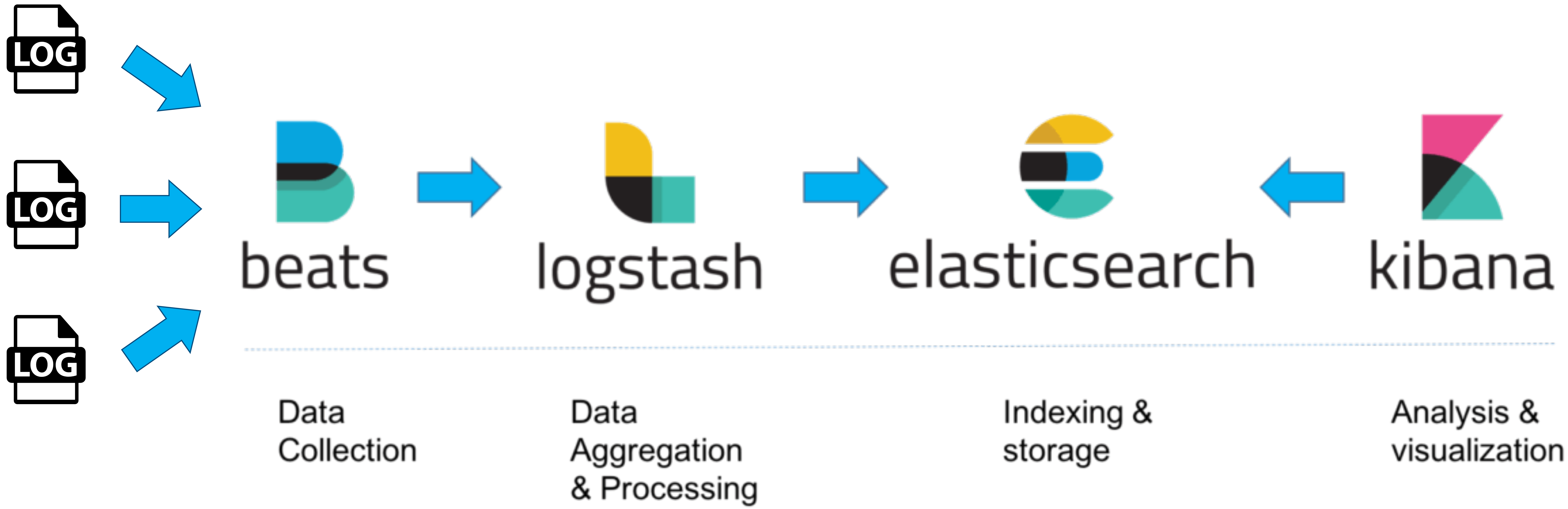
- execute SRM operations
- management of files and space
- authorisation permissions and interaction with Grid services.

One shared instance

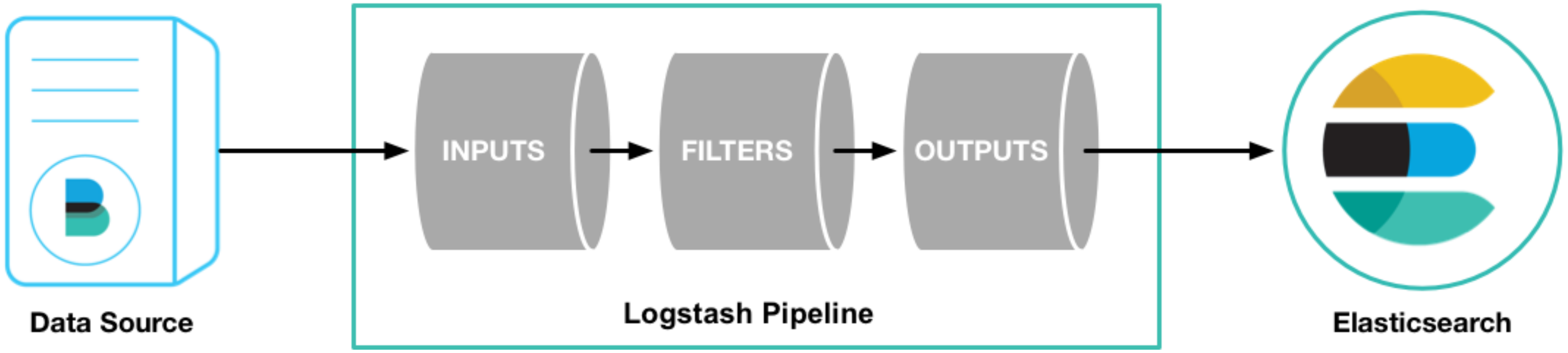
StoRM is currently adopted on several WLCG infrastructures, included INFN-CNAF Tier-1.



The ELK stack



Parse logs using Logstash



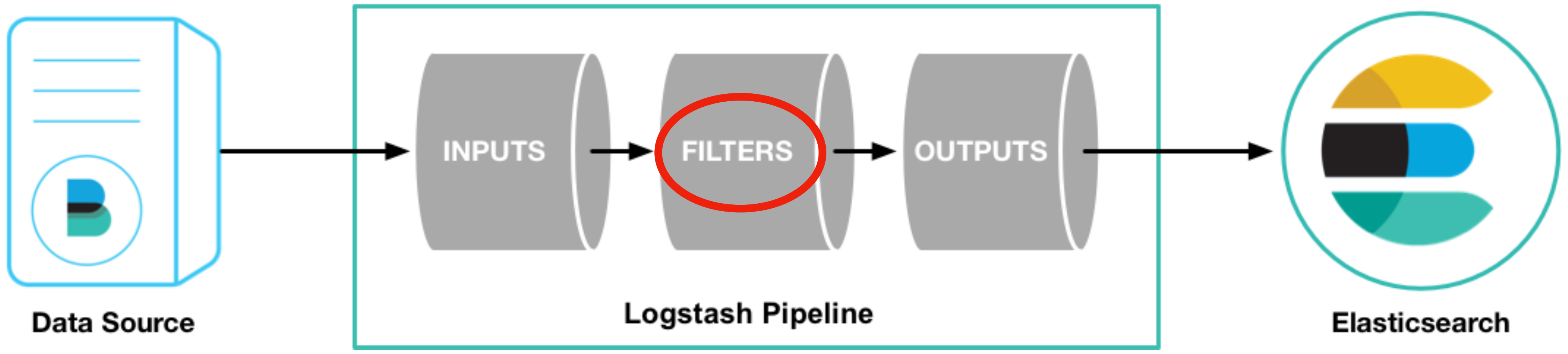
Inside the local cluster, Logstash creates a well defined pipeline.

- Input configuration that collects data from Beats in a continuous live-feed streaming.
- Filter configuration required for parsing each event, identify named fields to build a user defined structure.
- Output configuration to route parsed data in a search analytics engine (Elasticsearch).

Parse logs using Logstash

The different choice of filters for a correct parsing of log data is crucial.

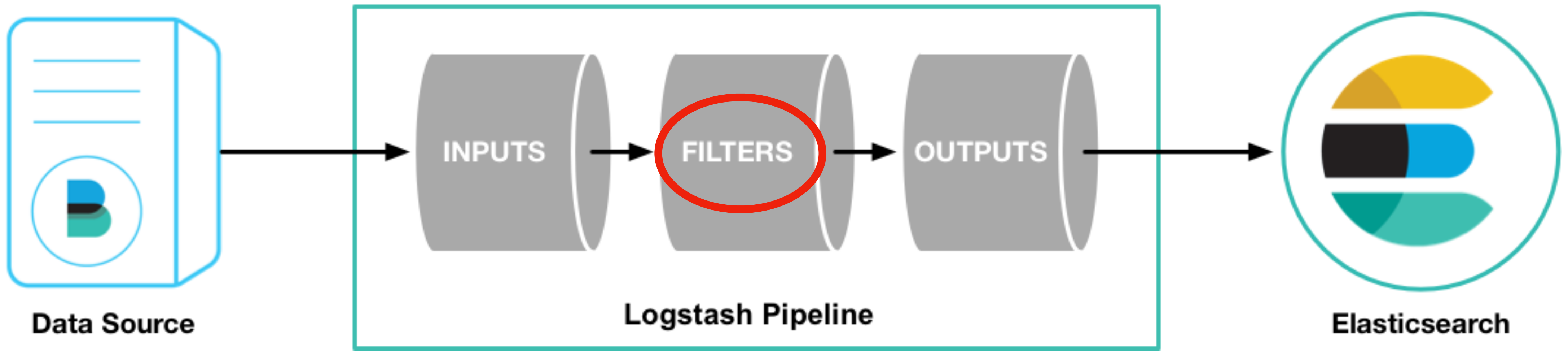
A large amount of them was parsed using the ***grok*** filter.



Parse logs using Logstash

The different choice of filters for a correct parsing of log data is crucial.

A large amount of them was parsed using the ***grok*** filter.



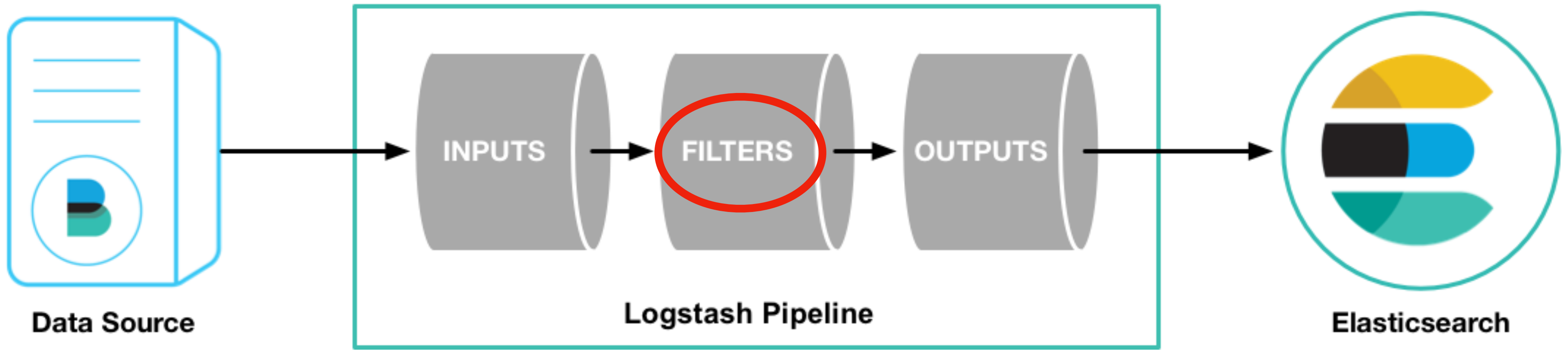
A grok filter, based on Regular Expressions, is adopted to match specific portions of log entries by creating a series of pattern defined as follows:

```
%{SYNTAX:SEMANTIC}
```

Parse logs using Logstash

The different choice of filters for a correct parsing of log data is crucial.

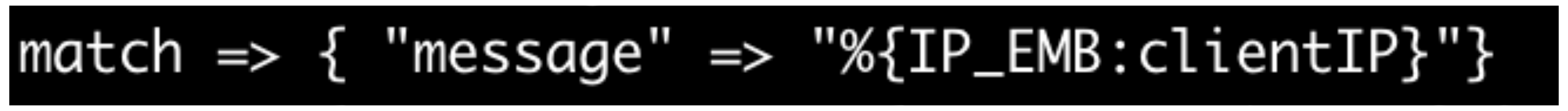
A large amount of them was parsed using the ***grok*** filter.



A grok filter, based on Regular Expressions, is adopted to match specific portions of log entries by creating a series of pattern defined as follows:

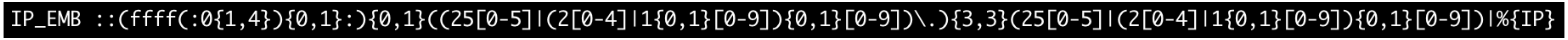
%{SYNTAX:SEMANTIC}

where SYNTAX is the name of the pattern that will match the text, while the SEMANTIC is the identifier of the piece of text being matched.



```
match => { "message" => "%{IP_EMB:clientIP}"}
```

Several patterns are predefined e.g. DATE, TIME. However, ***custom patterns*** are required in order to match every possible scenario. (Such patterns are stored in a specific file).



```
IP_EMB ::(ffff(:0{1,4}){0,1}:){0,1}((25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])\.){3,3}(25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])|%{IP}
```

Types of log parsed

Using Beats, several logs were parsed, coming from the ATLAS* application of StoRM instances.

- ✓ *storm-atlas*
  - ✓ storm-frontend-server.log
  - ✓ storm-backend.log
  - ✓ heartbeat.log
  - ✓ monitoring.log

→ Cluster containing both one frontend and the entire backend instances.

- ✓ *storm-fe-atlas-07*
  - ✓ storm-frontend-server.log
  - ✓ monitoring.log

→ Cluster containing only one frontend instance.

* ATLAS logs used as example.

Types of log parsed

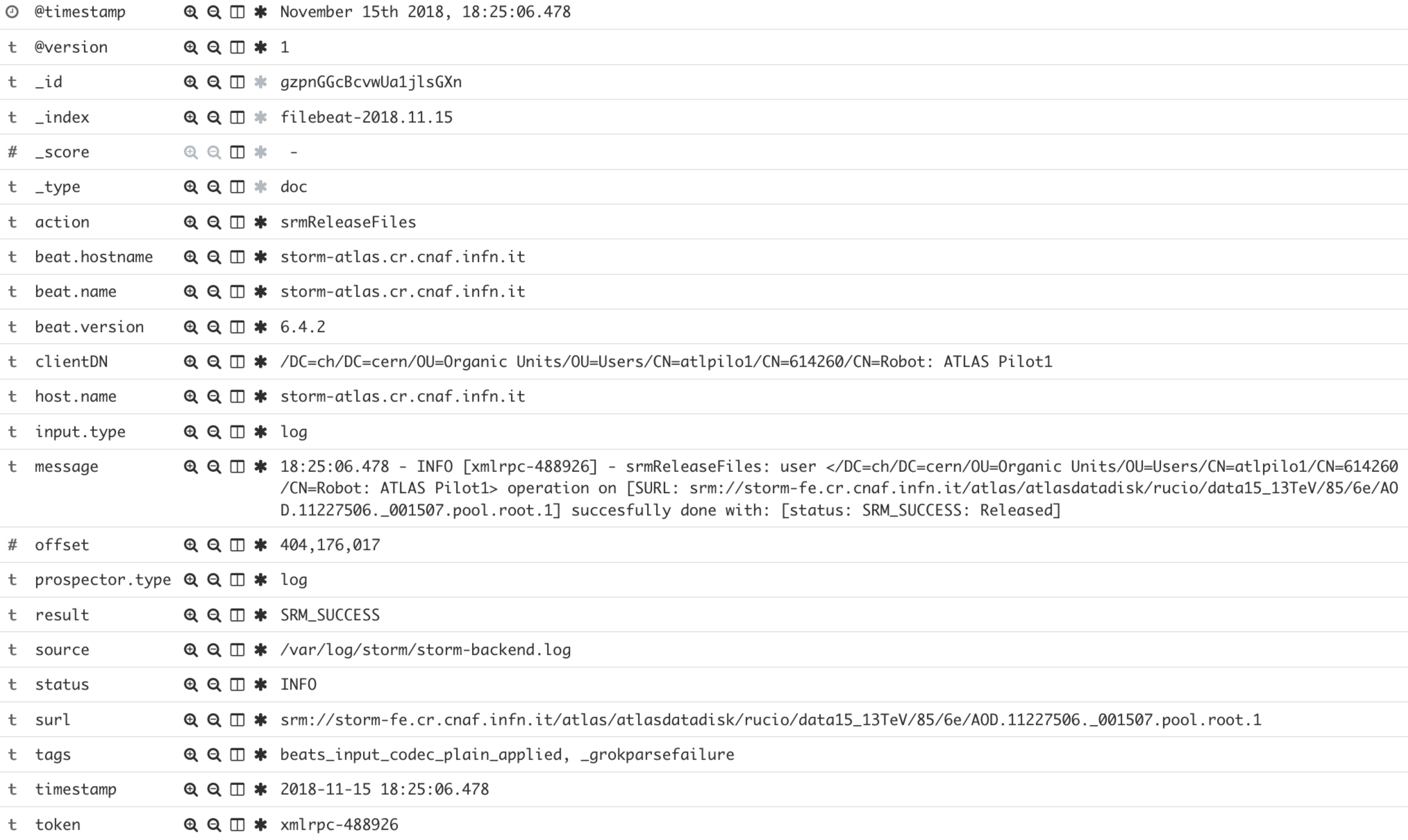
Using Beats, several logs were parsed, coming from the ATLAS* application of StoRM instances.

- ✓ *storm-atlas*
  - ✓ storm-frontend-server.log
  - ✓ storm-backend.log
  - ✓ heartbeat.log
  - ✓ monitoring.log
- ✓ *storm-fe-atlas-07*
  - ✓ storm-frontend-server.log
  - ✓ monitoring.log

→ All with a different structure and formalism!

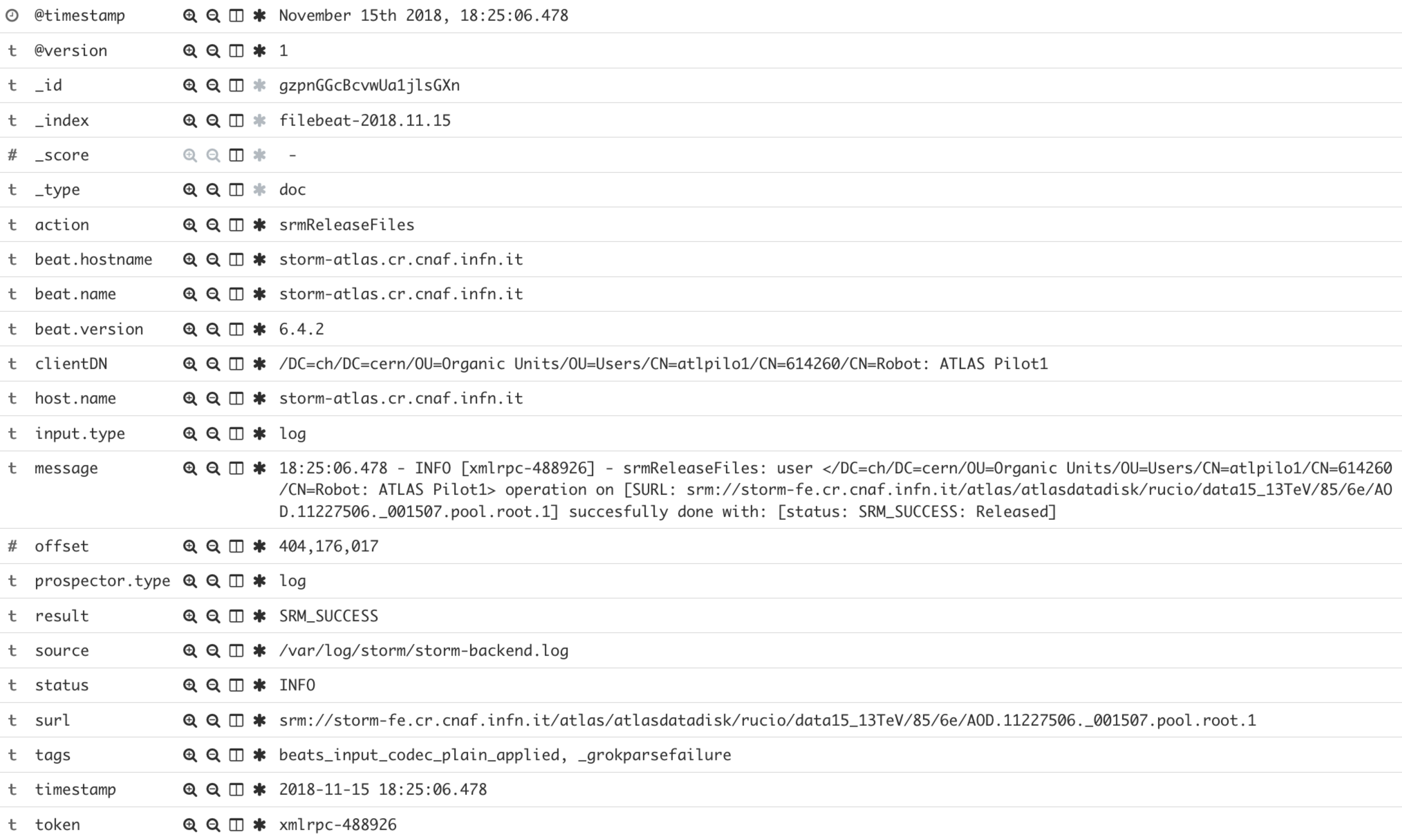
* ATLAS logs used as example.

Example of parsed log, with new structured information:



⏲	@timestamp	November 15th 2018, 18:25:06.478
t	@version	1
t	_id	gzpnGGcBcvwUa1jlsGXn
t	_index	filebeat-2018.11.15
#	_score	-
t	_type	doc
t	action	srmReleaseFiles
t	beat.hostname	storm-atlas.cr.cnaf.infn.it
t	beat.name	storm-atlas.cr.cnaf.infn.it
t	beat.version	6.4.2
t	clientDN	/DC=ch/DC=cern/OU=Organic Units/OU=Users/CN=atlpilo1/CN=614260/CN=Robot: ATLAS Pilot1
t	host.name	storm-atlas.cr.cnaf.infn.it
t	input.type	log
t	message	18:25:06.478 - INFO [xmlrpc-488926] - srmReleaseFiles: user </DC=ch/DC=cern/OU=Organic Units/OU=Users/CN=atlpilo1/CN=614260/CN=Robot: ATLAS Pilot1> operation on [SURL: srm://storm-fe.cr.cnaf.infn.it/atlas/atlasdatadisk/rucio/data15_13TeV/85/6e/AOD.11227506._001507.pool.root.1] succesfully done with: [status: SRM_SUCCESS: Released]
#	offset	404,176,017
t	prospector.type	log
t	result	SRM_SUCCESS
t	source	/var/log/storm/storm-backend.log
t	status	INFO
t	surl	srm://storm-fe.cr.cnaf.infn.it/atlas/atlasdatadisk/rucio/data15_13TeV/85/6e/AOD.11227506._001507.pool.root.1
t	tags	beats_input_codec_plain_applied, _grokparsefailure
t	timestamp	2018-11-15 18:25:06.478
t	token	xmlrpc-488926

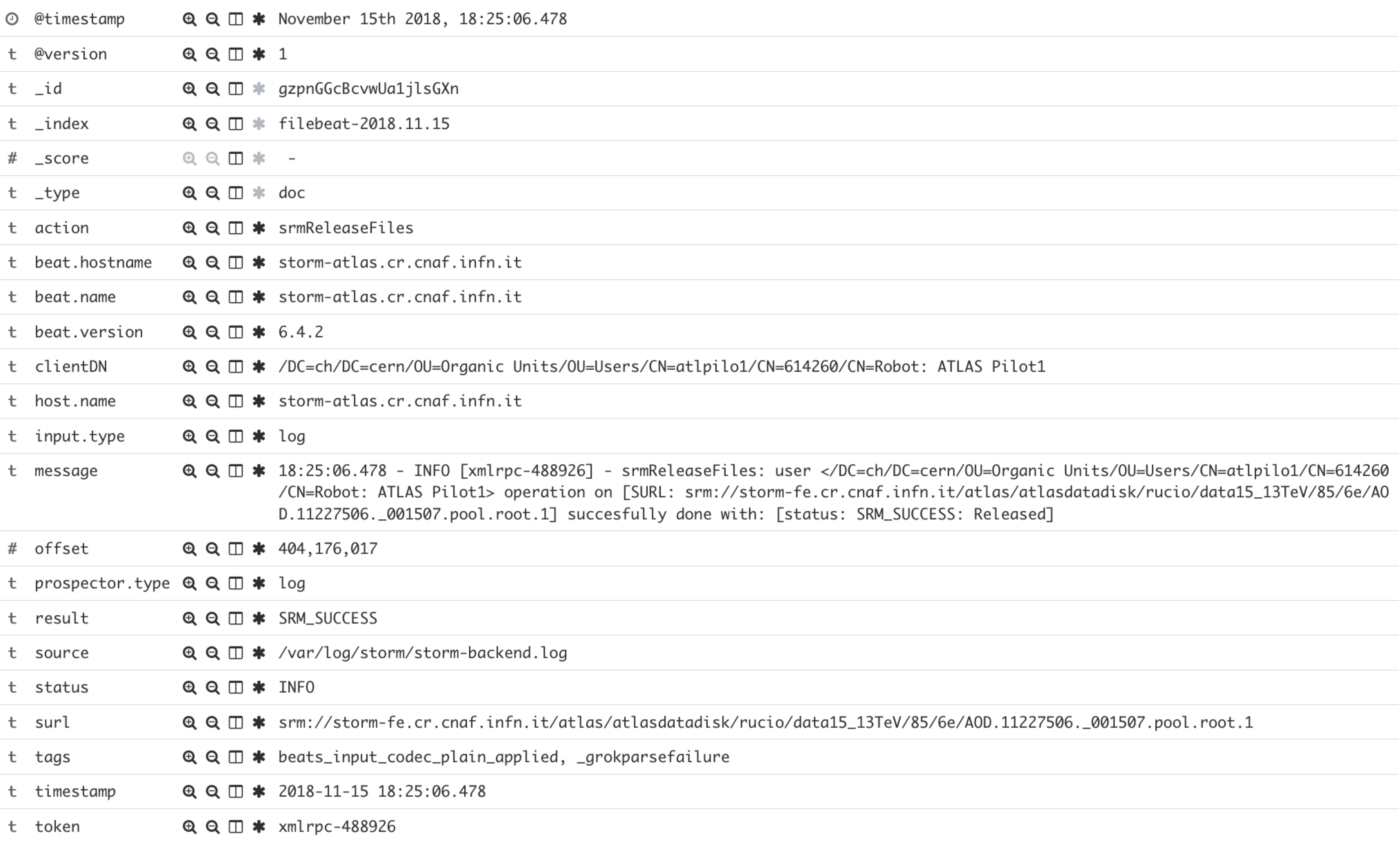
Example of parsed log, with new structured information:



	@timestamp	November 15th 2018, 18:25:06.478
t	@version	1
t	_id	gzpnGGcBcvwUa1jlsGXn
t	_index	filebeat-2018.11.15
#	_score	-
t	_type	doc
t	action	srmReleaseFiles
t	beat.hostname	storm-atlas.cr.cnaf.infn.it
t	beat.name	storm-atlas.cr.cnaf.infn.it
t	beat.version	6.4.2
t	clientDN	/DC=ch/DC=cern/OU=Organic Units/OU=Users/CN=atlpilo1/CN=614260/CN=Robot: ATLAS Pilot1
t	host.name	storm-atlas.cr.cnaf.infn.it
t	input.type	log
t	message	18:25:06.478 - INFO [xmlrpc-488926] - srmReleaseFiles: user </DC=ch/DC=cern/OU=Organic Units/OU=Users/CN=atlpilo1/CN=614260/CN=Robot: ATLAS Pilot1> operation on [SURL: srm://storm-fe.cr.cnaf.infn.it/atlas/atlasdatadisk/rucio/data15_13TeV/85/6e/AOD.11227506._001507.pool.root.1] succesfully done with: [status: SRM_SUCCESS: Released]
#	offset	404,176,017
t	prospector.type	log
t	result	SRM_SUCCESS
t	source	/var/log/storm/storm-backend.log
t	status	INFO
t	surl	srm://storm-fe.cr.cnaf.infn.it/atlas/atlasdatadisk/rucio/data15_13TeV/85/6e/AOD.11227506._001507.pool.root.1
t	tags	beats_input_codec_plain_applied, _grokparsefailure
t	timestamp	2018-11-15 18:25:06.478
t	token	xmlrpc-488926

→ Original message (remains in the log document)

Example of parsed log, with new structured information:



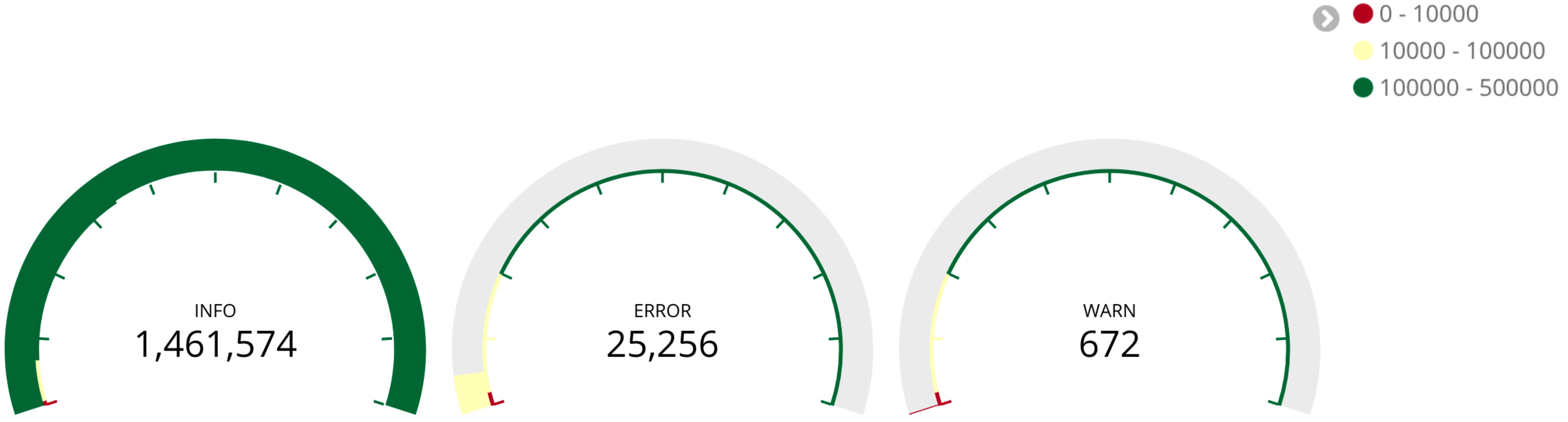
◷	@timestamp	November 15th 2018, 18:25:06.478
t	@version	1
t	_id	gzpnGGcBcvwUa1jlsGXn
t	_index	filebeat-2018.11.15
#	_score	-
t	_type	doc
t	action	srmReleaseFiles
t	beat.hostname	storm-atlas.cr.cnaf.infn.it
t	beat.name	storm-atlas.cr.cnaf.infn.it
t	beat.version	6.4.2
t	clientDN	/DC=ch/DC=cern/OU=Organic Units/OU=Users/CN=atlpilo1/CN=614260/CN=Robot: ATLAS Pilot1
t	host.name	storm-atlas.cr.cnaf.infn.it
t	input.type	log
t	message	18:25:06.478 - INFO [xmlrpc-488926] - srmReleaseFiles: user </DC=ch/DC=cern/OU=Organic Units/OU=Users/CN=atlpilo1/CN=614260/CN=Robot: ATLAS Pilot1> operation on [SURL: srm://storm-fe.cr.cnaf.infn.it/atlas/atlasdatadisk/rucio/data15_13TeV/85/6e/AOD.11227506._001507.pool.root.1] succesfully done with: [status: SRM_SUCCESS: Released]
#	offset	404,176,017
t	prospector.type	log
t	result	SRM_SUCCESS
t	source	/var/log/storm/storm-backend.log
t	status	INFO
t	surl	srm://storm-fe.cr.cnaf.infn.it/atlas/atlasdatadisk/rucio/data15_13TeV/85/6e/AOD.11227506._001507.pool.root.1
t	tags	beats_input_codec_plain_applied, _grokparsefailure
t	timestamp	2018-11-15 18:25:06.478
t	token	xmlrpc-488926

→ Timestamp, in a date specific format.

→ Original message (remains in the log document)

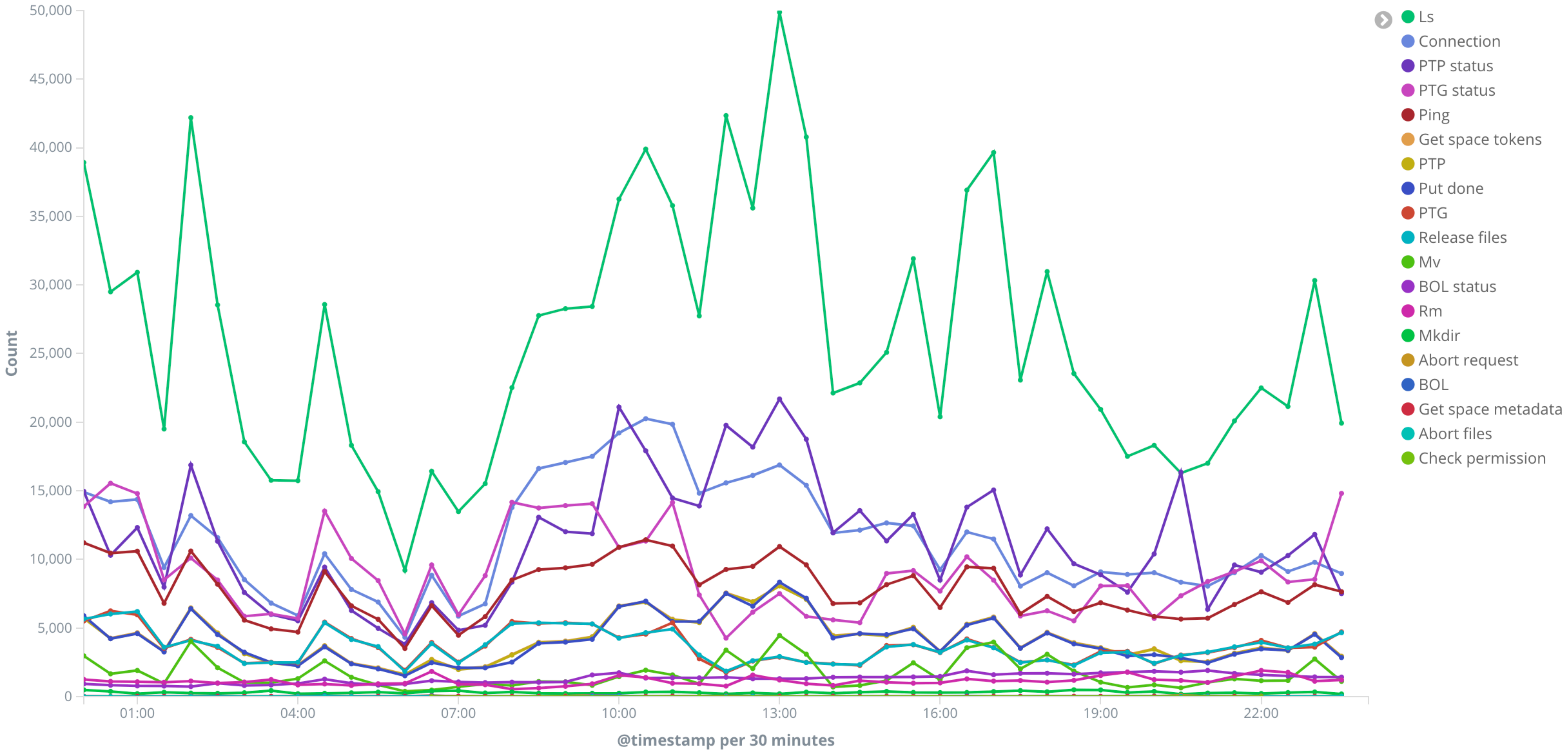
Visualize data

Using Kibana User Interface, it is possible to create new visualisations and collect them to form new dashboards.
Example: 1 day of logs (25th of November 2018 - UTC).

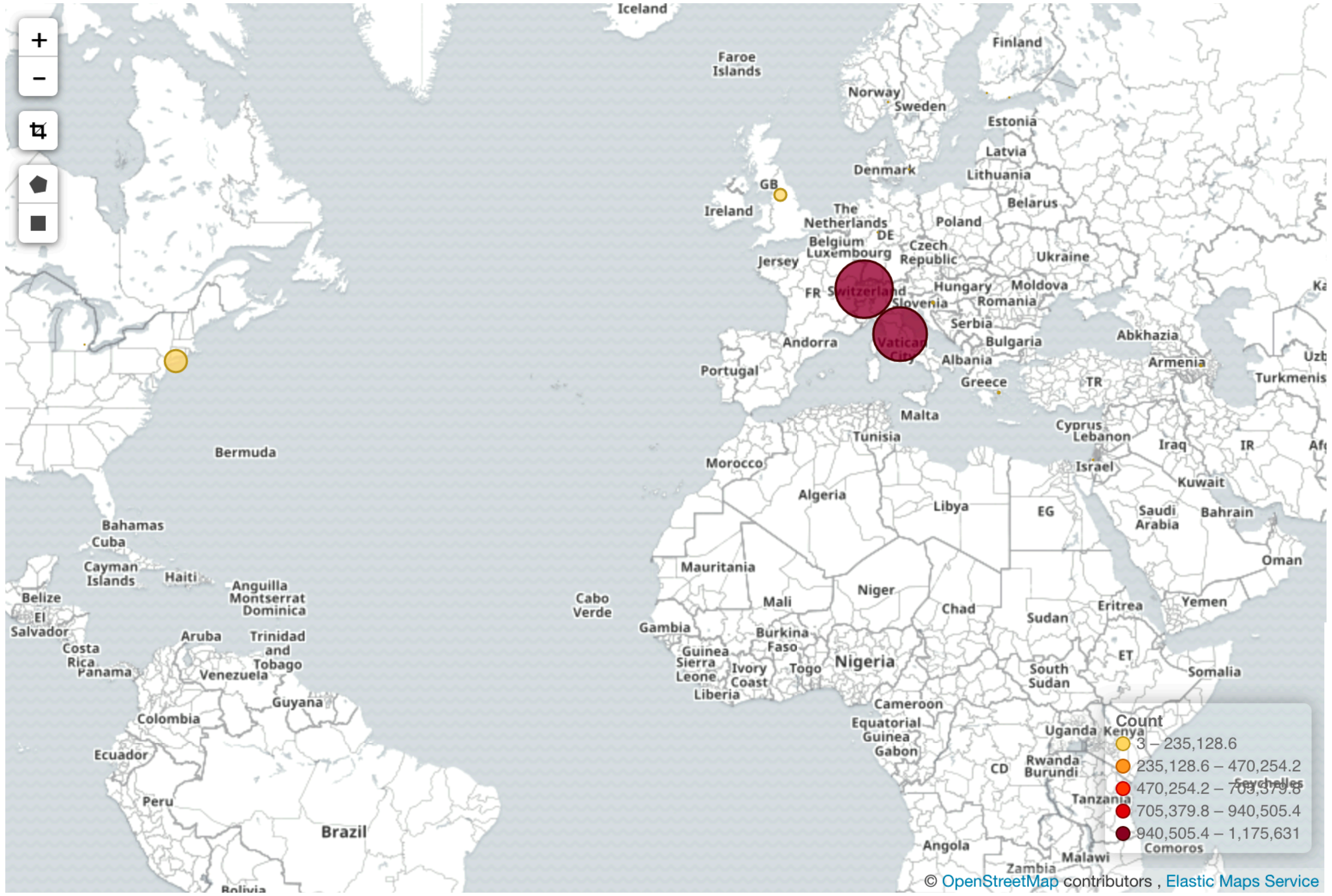


Count of INFO, ERROR and WARN logs from the StoRM Back-End instance.

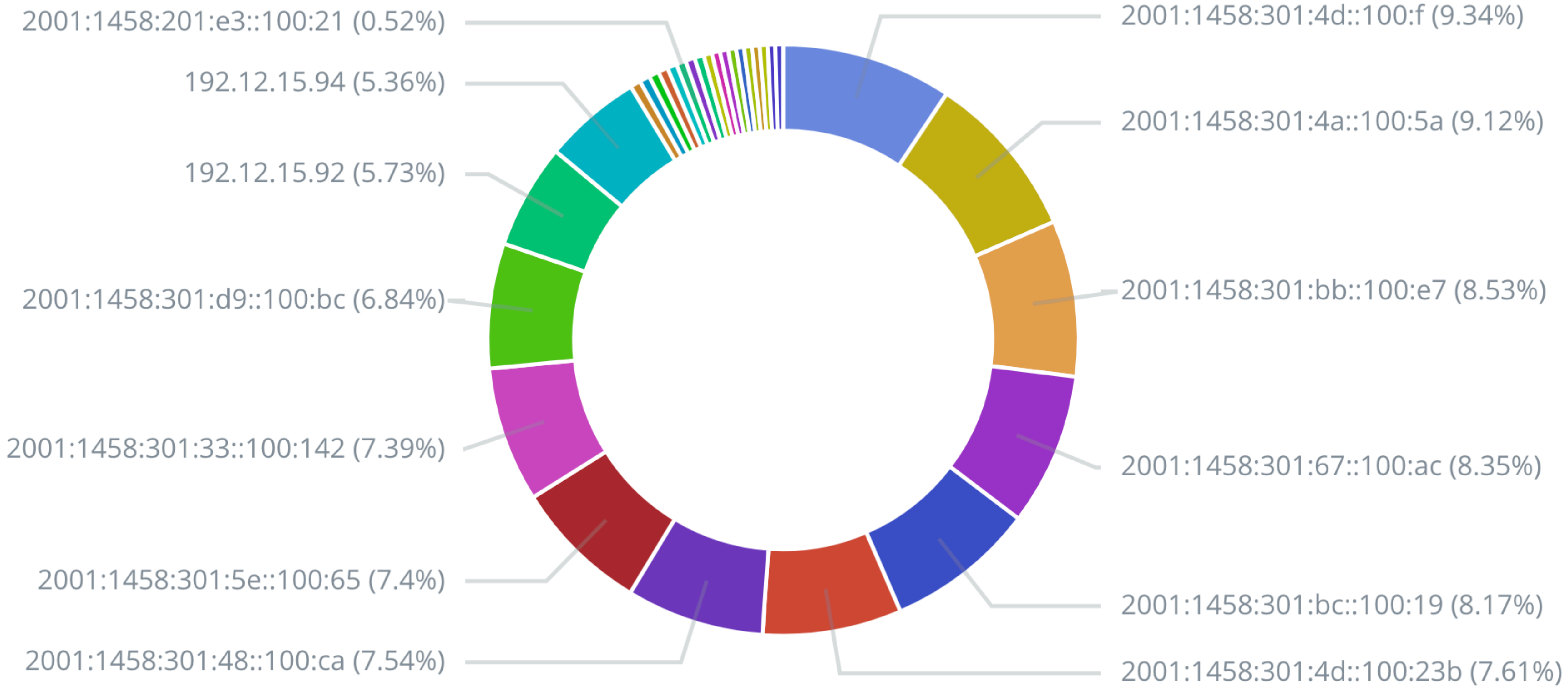
Visualize data



Count of different operations for the StoRM Front-End. (the Back-End similar plot is not shown.)



Map of client IP addresses location and frequency of the top 30 (not more for visualisation purposes).

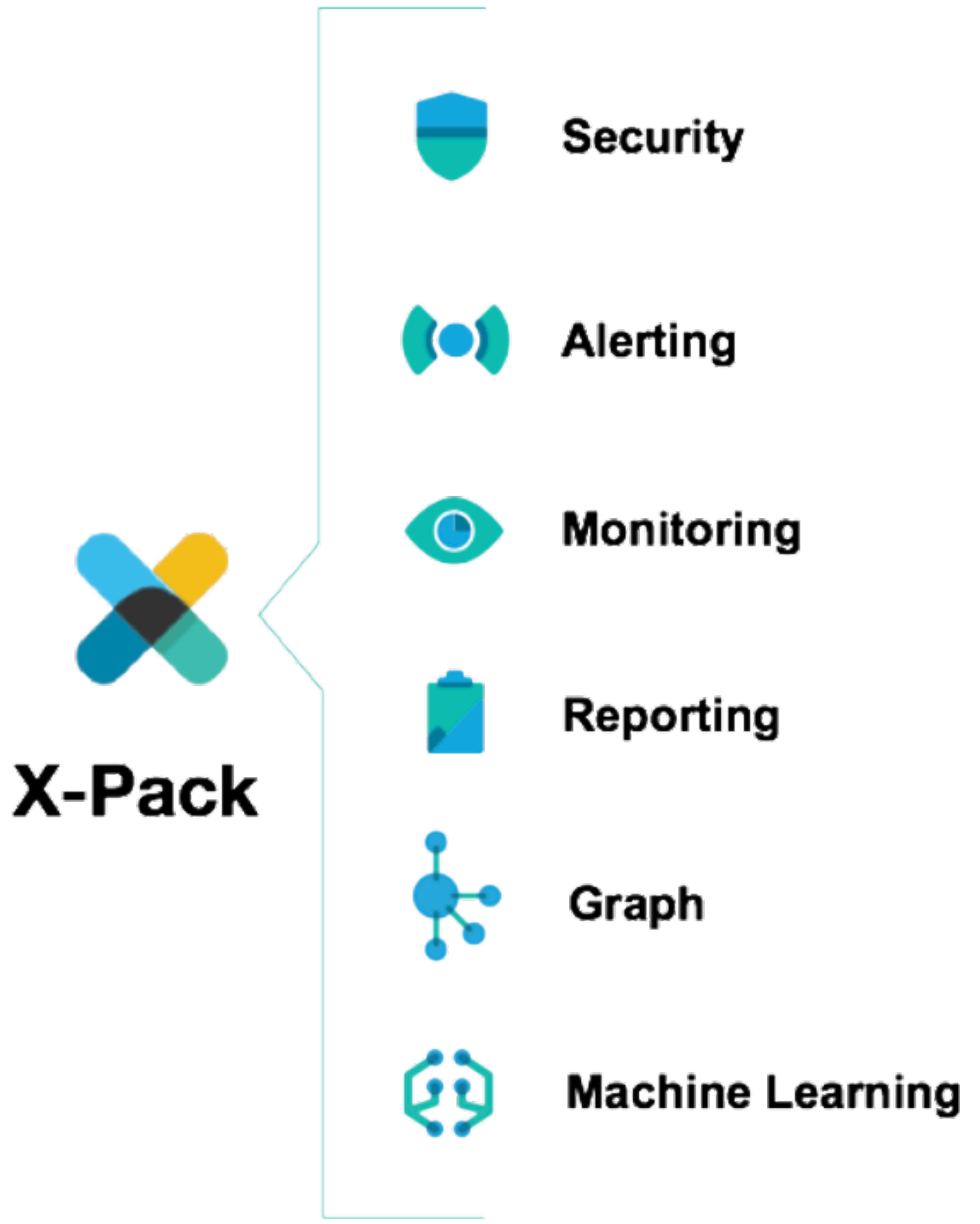


Machine Learning analytics

With the new Elasticsearch major release, among the **premium functionalities** provided with X-Pack, Machine Learning capabilities for data analysis were added.

Using **proprietary** unsupervised learning techniques, this functionality is mainly used for **anomaly detection** use cases.

If the anomalies found are interesting, it is then possible to create a real time anomaly alerting system for operators and experts.



X-Pack

Security

Alerting

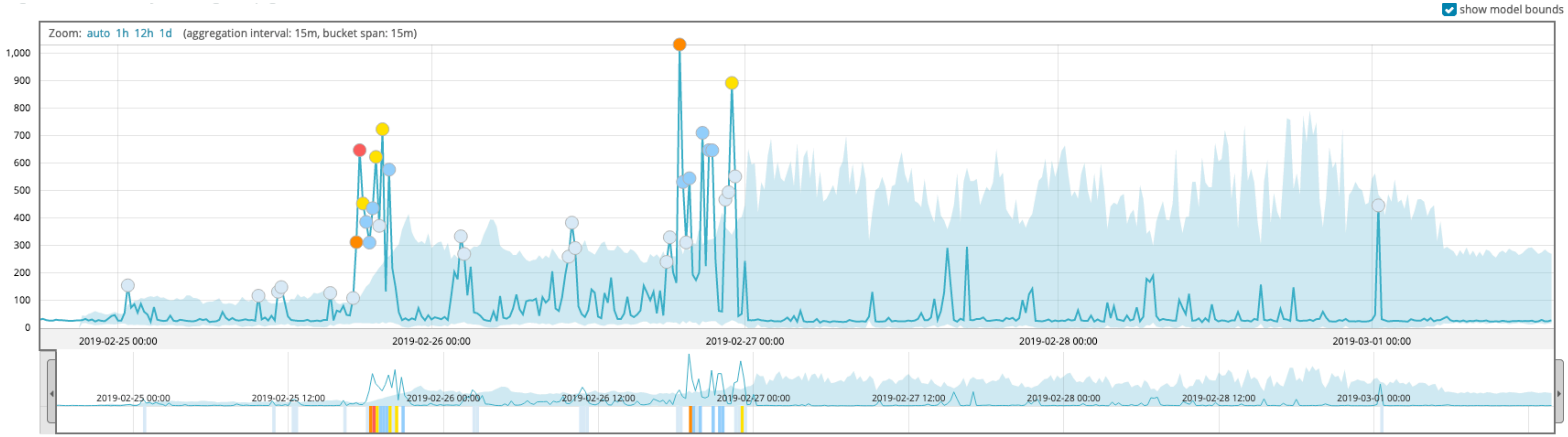
Monitoring

Reporting

Graph

Machine Learning

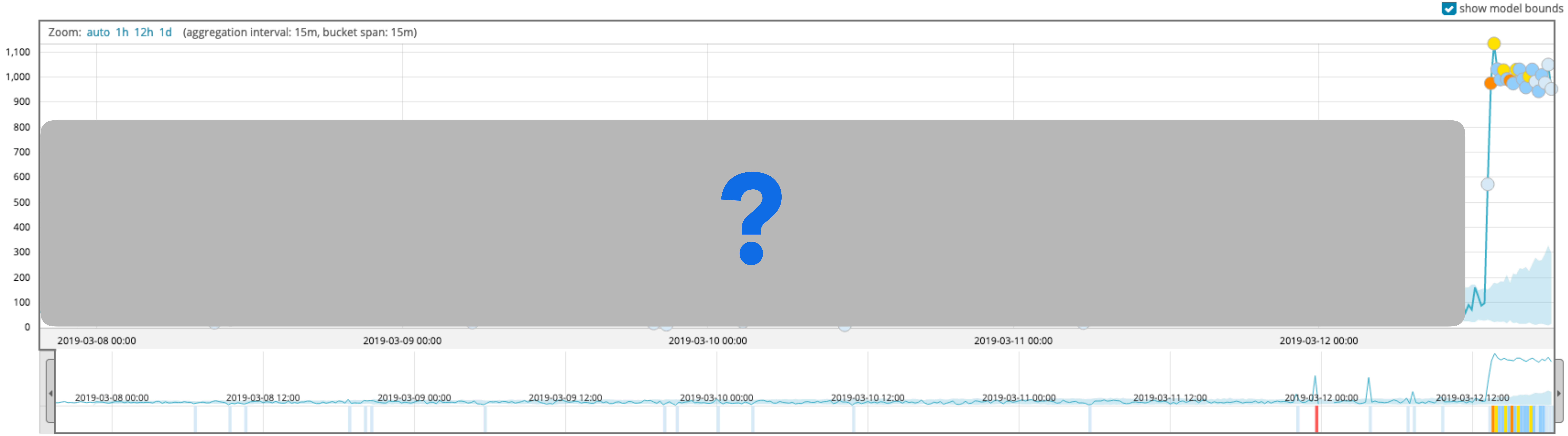
Machine Learning analytics



Duration in milliseconds of the last bunch of *Prepare To Get* StoRM operations.
(from *heartbeat.log* of StoRM Backend)

Machine Learning analytics

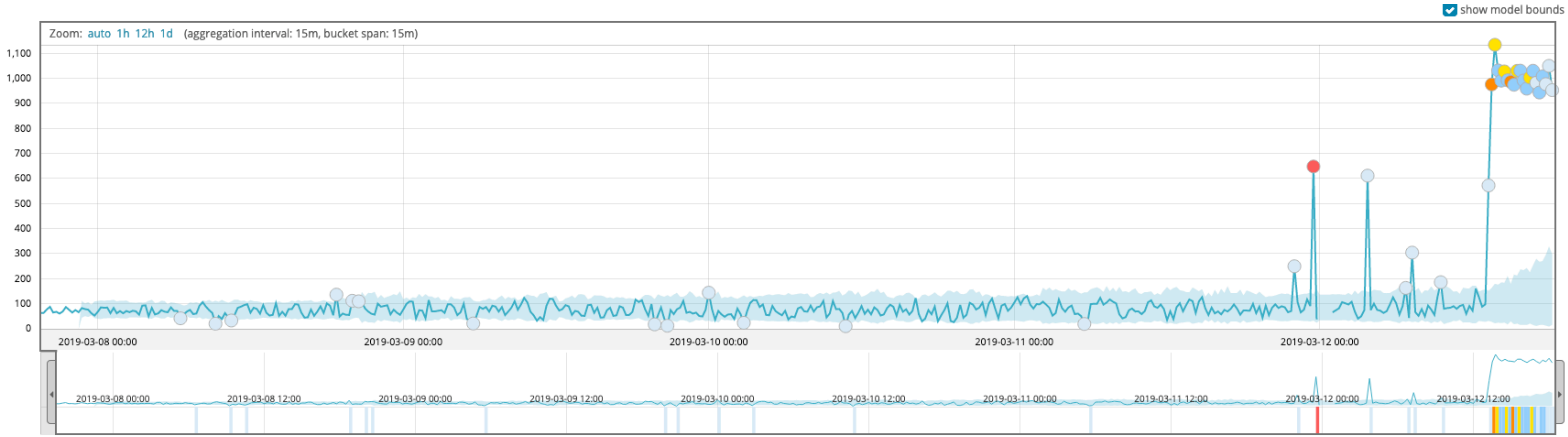
What about **predictions**? Is this tool capable of anticipate any potential issue?



Average duration of the latest bunch of operation on the StoRM backend service.

Machine Learning analytics

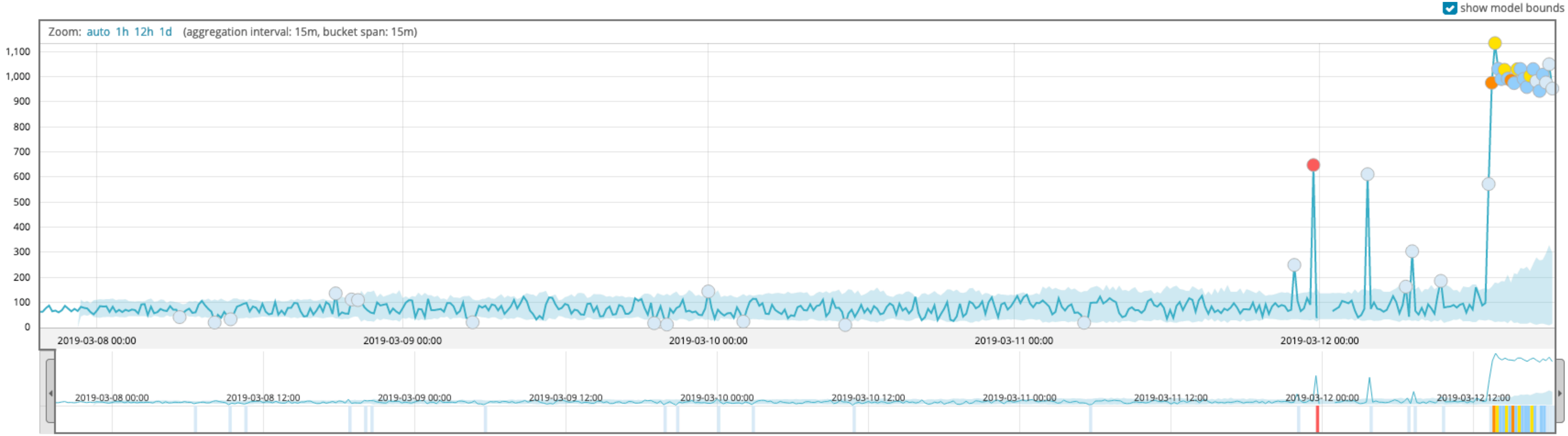
What about **predictions**? Is this tool capable of anticipate any potential issue?



Machine Learning analytics

What about **predictions**? Is this tool capable of anticipate any potential issue?

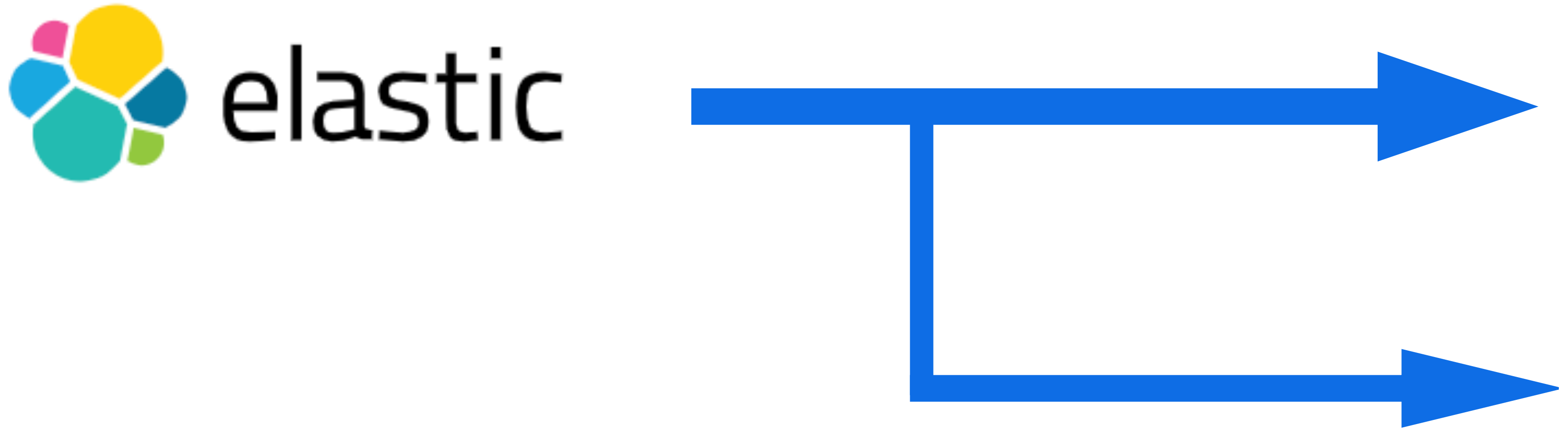
The system, however, is not designed for this kind of operation.



Zoom: auto 1h 12h 1d (aggregation interval: 15m, bucket span: 15m)

Future Steps @CNAF

- Centralise logs on a storage partition of the Tier-1. A unique *log* file with all the information appended from different services.
    - One NFS mountpoint to allow data read from any VM.
- Cluster for ELK on a dedicated Tier-1 physical cluster.



Apply ML algorithms, outside Elastic, using conventional framework

[Parallel on *Data Management and Big Data*, by Luca Giommi]

Use analytics engine for Big Data processing, like **Apache Spark**

- Spark cluster on a cloud machine at CNAF with 3 storage volumes of 300GB each.

[Installed using DODAS (Indigo)]

- Other logs also taken in consideration: WNs, service machines, gpfs, gridftp, xrootd, batch system and application level logging.

Conclusions

Using the ELK Stack, it was possible to create a centralised platform for logs coming from the StoRM service at CNAF.

Using a premium functionality of this suite, a Machine Learning approach on such logs was adopted in particular for an *anomaly detection* use-case.

Despite useful for on-line analytics and monitoring, this may not be the optimal solution for a predictive scenario and a proactive identification of failures.

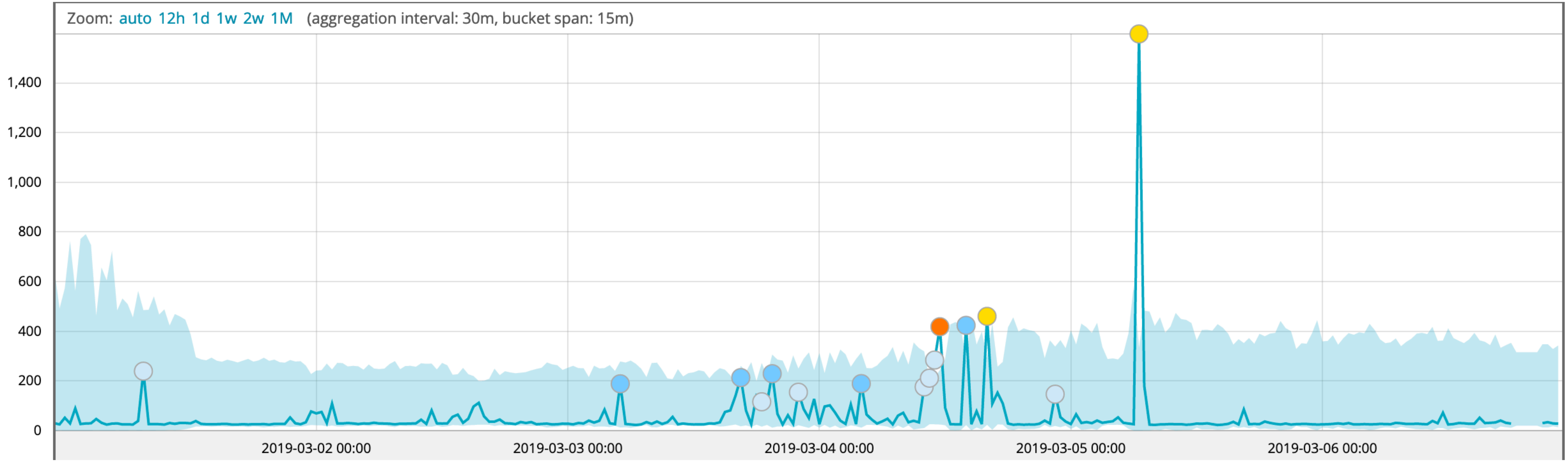
Moving in this direction, new approaches are being investigated at CNAF, such as the implementation of a Spark cluster for a Big Data oriented analysis.

Thank you!

For information and contacts:

Backup Slides

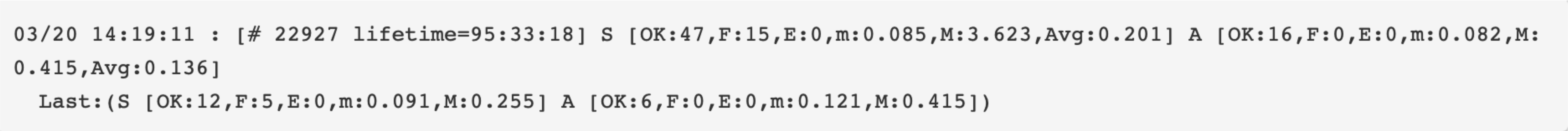
Other example



Zoom: auto 12h 1d 1w 2w 1M (aggregation interval: 30m, bucket span: 15m)

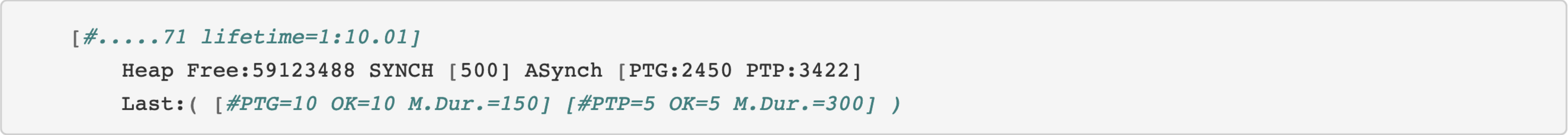
Duration of the last bunch of ptg operations in the *storm-backend-metrics.log*

monitoring.log line



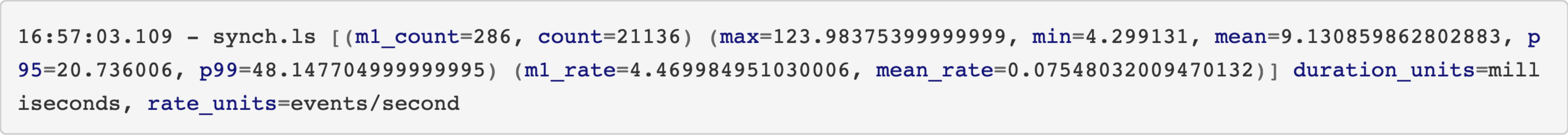
```
03/20 14:19:11 : [# 22927 lifetime=95:33:18] S [OK:47,F:15,E:0,m:0.085,M:3.623,Avg:0.201] A [OK:16,F:0,E:0,m:0.082,M:
0.415,Avg:0.136]
  Last:(S [OK:12,F:5,E:0,m:0.091,M:0.255] A [OK:6,F:0,E:0,m:0.121,M:0.415])
```

heartbeat.*log* line



```
[#.....71 lifetime=1:10.01]
   Heap Free:59123488 SYNCH [500] ASynch [PTG:2450 PTP:3422]
   Last:( [#PTG=10 OK=10 M.Dur.=150] [#PTP=5 OK=5 M.Dur.=300] )
```

storm-*backend-metrics.log* line



```
16:57:03.109 - synch.ls [(m1_count=286, count=21136) (max=123.98375399999999, min=4.299131, mean=9.130859862802883, p95=20.736006, p99=48.147704999999995) (m1_rate=4.469984951030006, mean_rate=0.07548032009470132)] duration_units=milliseconds, rate_units=events/second
```