Meltwater
Outside Insight

To a Billion Messages

and Beyond

János Csorvási

János Csorvási

Infrastructure Engineer at Meltwater

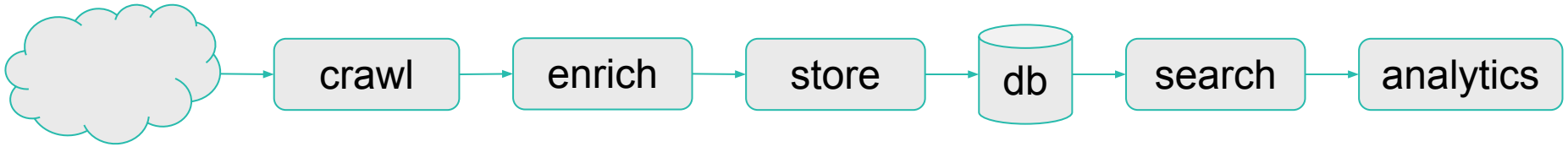
Engineering Enablement Mission

Meltwater

What is Engineering Enablement?

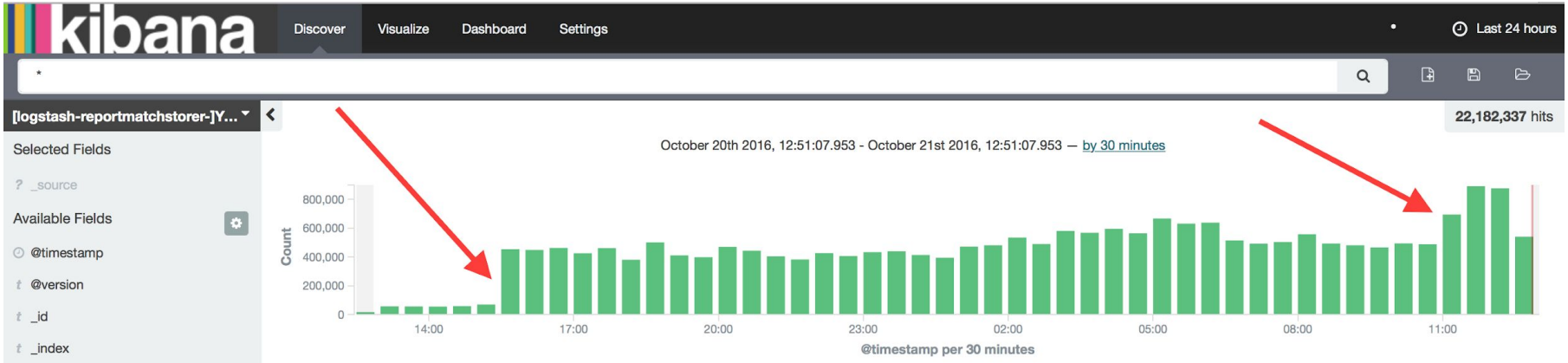
What is Meltwater?

Meltwater

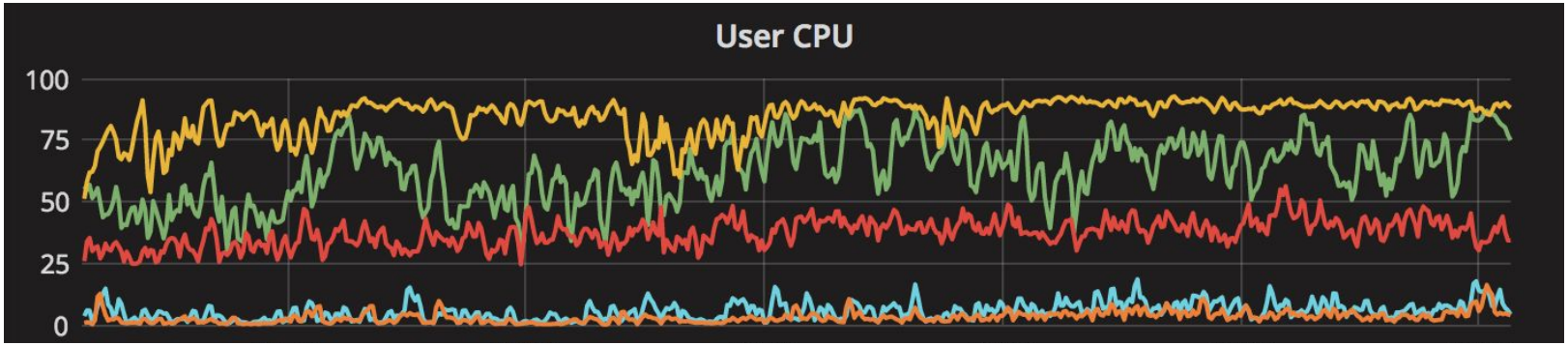


Meltwater

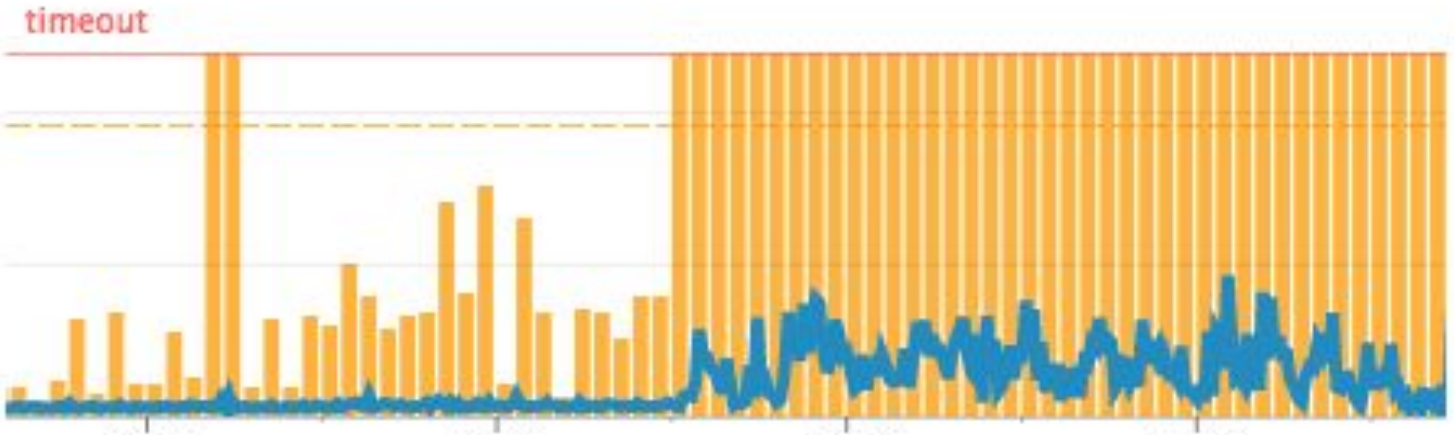
What is this talk about?


kibana
Discover
Visualize
Dashboard
Settings
Last 24 hours
*
[logstash-reportmatchstorer-]Y...
22,182,337 hits
Selected Fields
? _source
Available Fields
@timestamp
@version
_id
_index
October 20th 2016, 12:51:07.953 - October 21st 2016, 12:51:07.953 — by 30 minutes
Count
800,000
600,000
400,000
200,000
0
14:00
17:00
20:00
23:00
02:00
05:00
08:00
11:00
@timestamp per 30 minutes

Meltwater



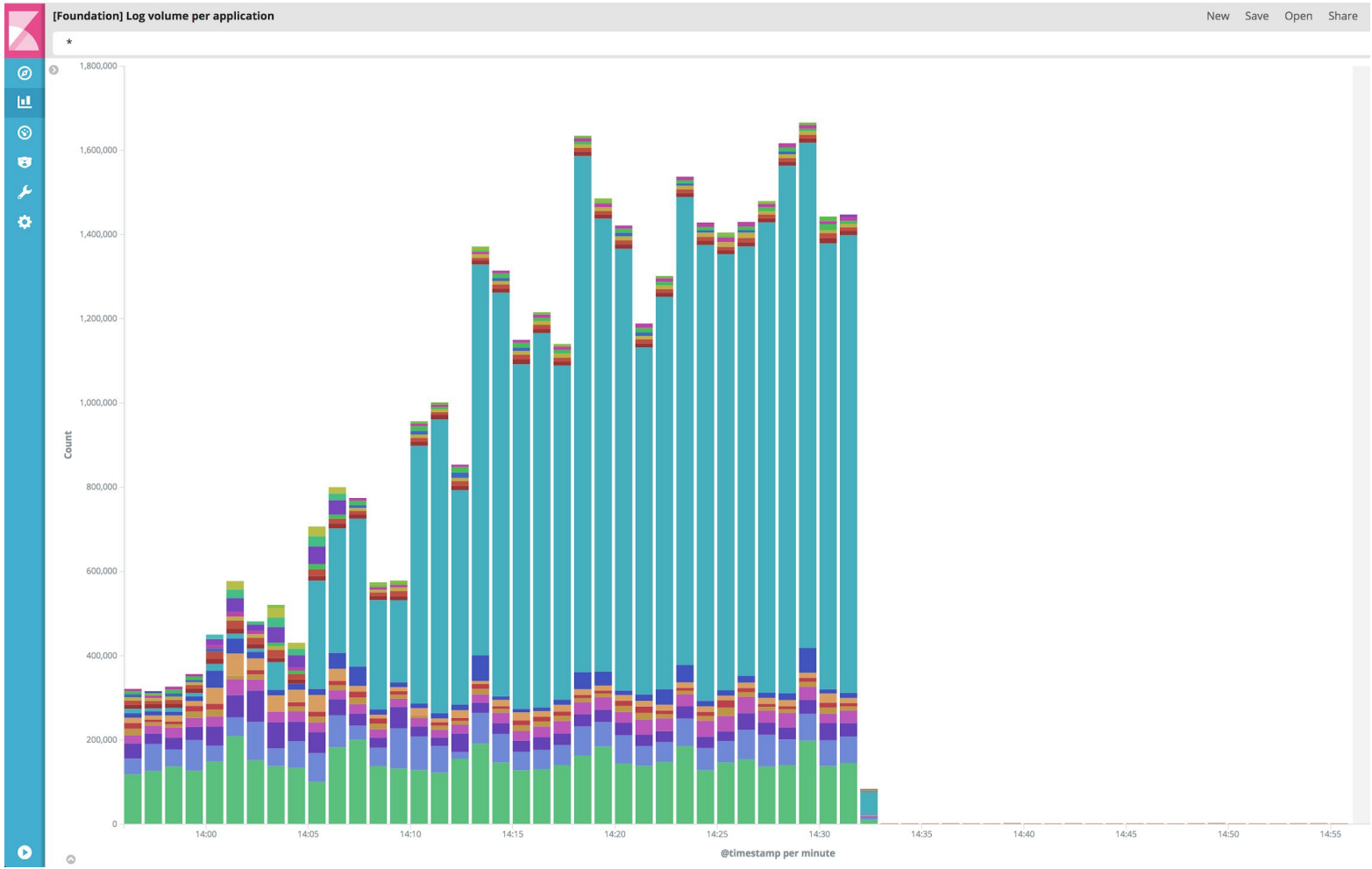
Meltwater



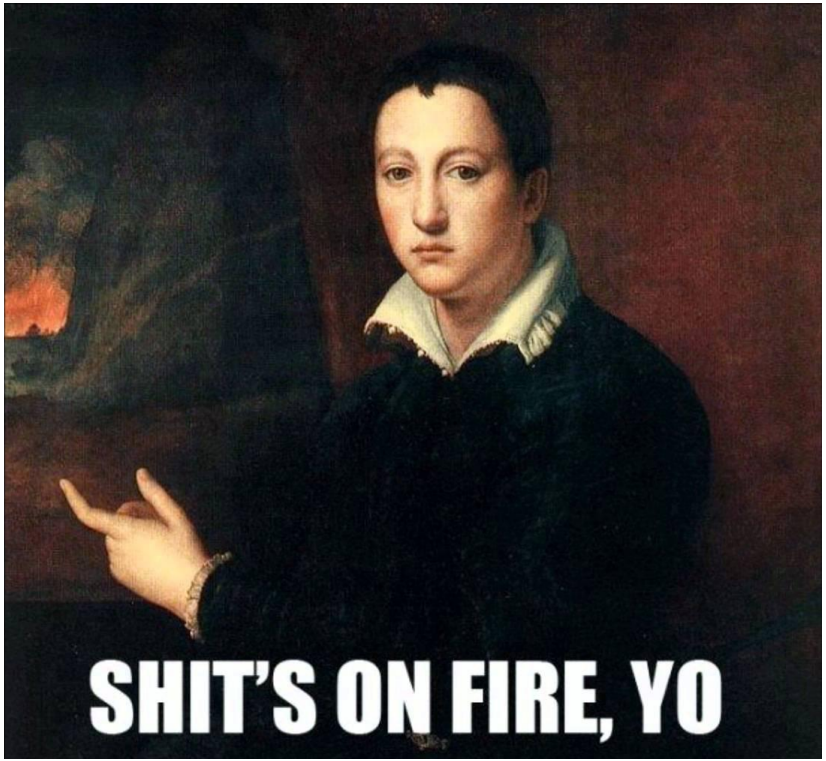
Meltwater

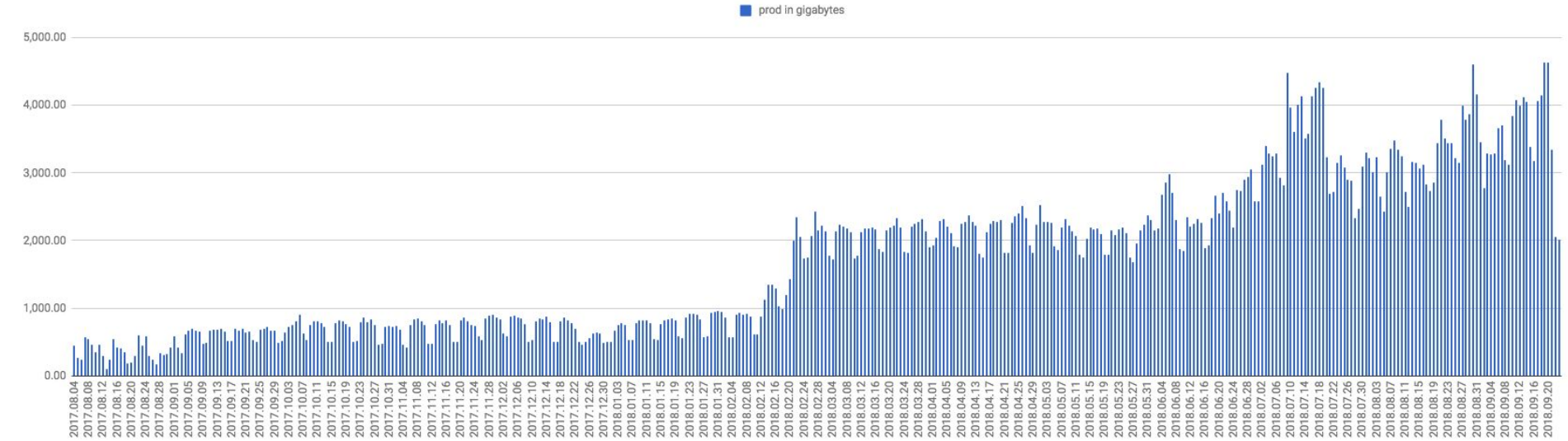
```
-       <root level="INFO">
+       <root level="TRACE">
                <appender-ref ref="LOGSTASH"/>
        </root>
```

```
for (Item item : items) {
    log.trace("Starting to process item {}", item);
    process(item);
    log.trace("Done processing item {}", item);
}
```


[Foundation] Log volume per application
New
Save
Open
Share
*
1,800,000
1,600,000
1,400,000
1,200,000
1,000,000
800,000
600,000
400,000
200,000
0
Count
14:00
14:05
14:10
14:15
14:20
14:25
14:30
14:35
14:40
14:45
14:50
14:55
@timestamp per minute

Meltwater




prod in gigabytes
5,000.00
4,000.00
3,000.00
2,000.00
1,000.00
0.00
2017.08.04
2017.08.08
2017.08.12
2017.08.16
2017.08.20
2017.08.24
2017.08.28
2017.09.01
2017.09.05
2017.09.09
2017.09.13
2017.09.17
2017.09.21
2017.09.25
2017.09.29
2017.10.03
2017.10.07
2017.10.11
2017.10.15
2017.10.19
2017.10.23
2017.10.27
2017.10.31
2017.11.04
2017.11.08
2017.11.12
2017.11.16
2017.11.20
2017.11.24
2017.11.28
2017.12.02
2017.12.06
2017.12.10
2017.12.14
2017.12.18
2017.12.22
2017.12.26
2017.12.30
2018.01.03
2018.01.07
2018.01.11
2018.01.15
2018.01.19
2018.01.23
2018.01.27
2018.01.31
2018.02.04
2018.02.08
2018.02.12
2018.02.16
2018.02.20
2018.02.24
2018.02.28
2018.03.04
2018.03.08
2018.03.12
2018.03.16
2018.03.20
2018.03.24
2018.03.28
2018.04.01
2018.04.05
2018.04.09
2018.04.13
2018.04.17
2018.04.21
2018.04.25
2018.04.29
2018.05.03
2018.05.07
2018.05.11
2018.05.15
2018.05.19
2018.05.23
2018.05.27
2018.05.31
2018.06.04
2018.06.08
2018.06.12
2018.06.16
2018.06.20
2018.06.24
2018.06.28
2018.07.02
2018.07.06
2018.07.10
2018.07.14
2018.07.18
2018.07.22
2018.07.26
2018.07.30
2018.08.03
2018.08.07
2018.08.11
2018.08.15
2018.08.19
2018.08.23
2018.08.27
2018.08.31
2018.09.04
2018.09.08
2018.09.12
2018.09.16
2018.09.20

Meltwater

3 800 GB / day (with replicas)

1 700 000 000 msg / day

40 000 msg / sec (with replicas)

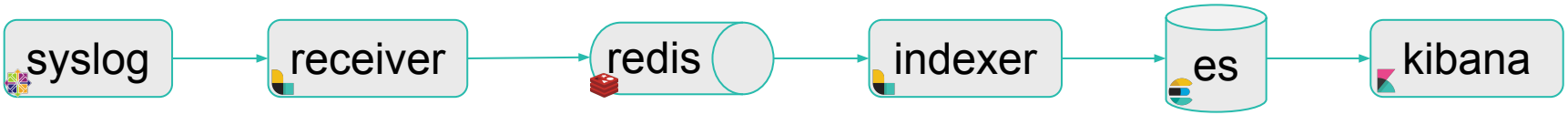
32 day rolling retention

Meltwater

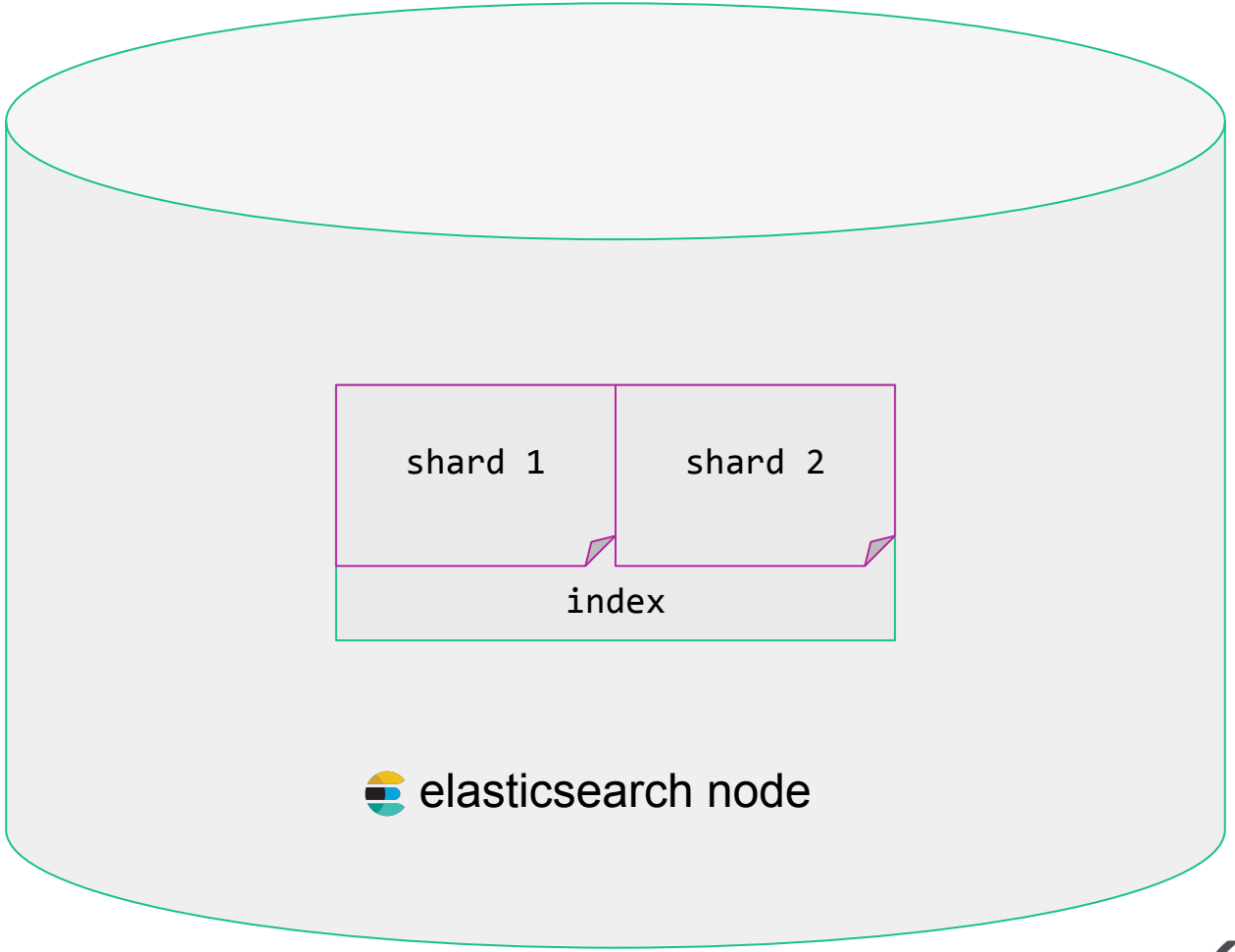
How did we evolve?

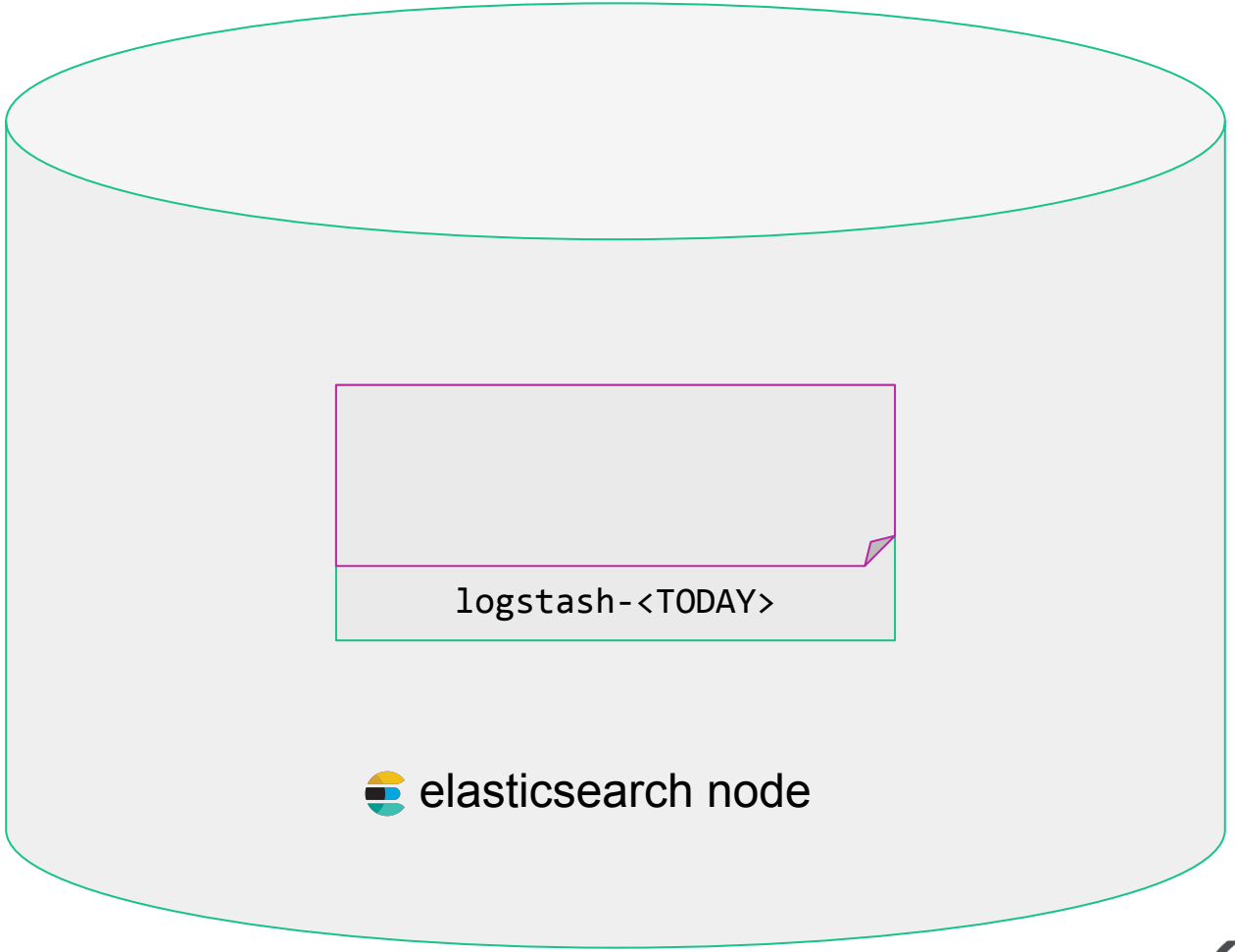
Inherited system

Meltwater

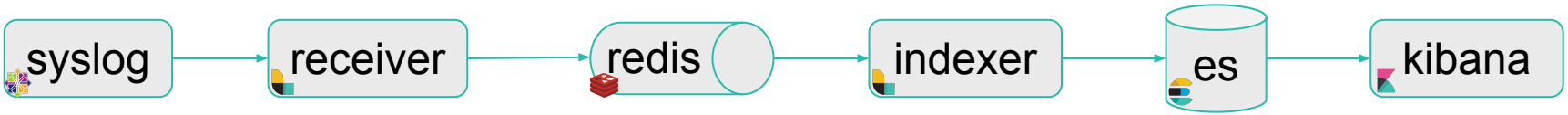


Meltwater

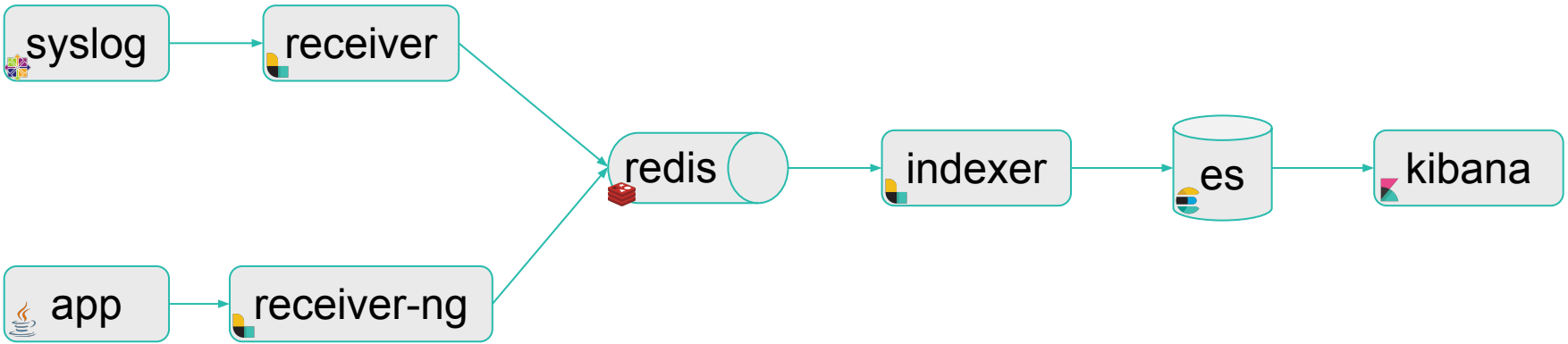


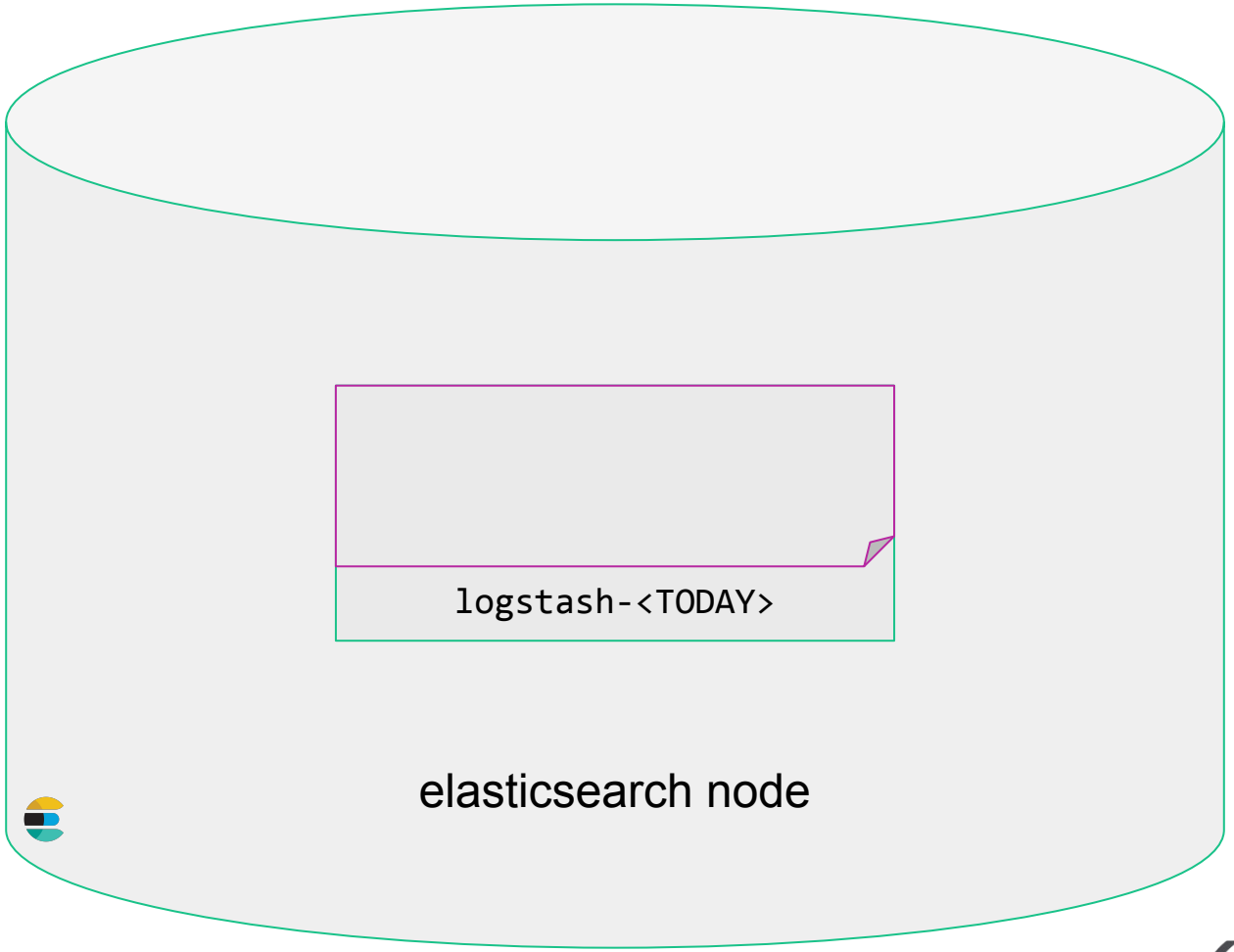


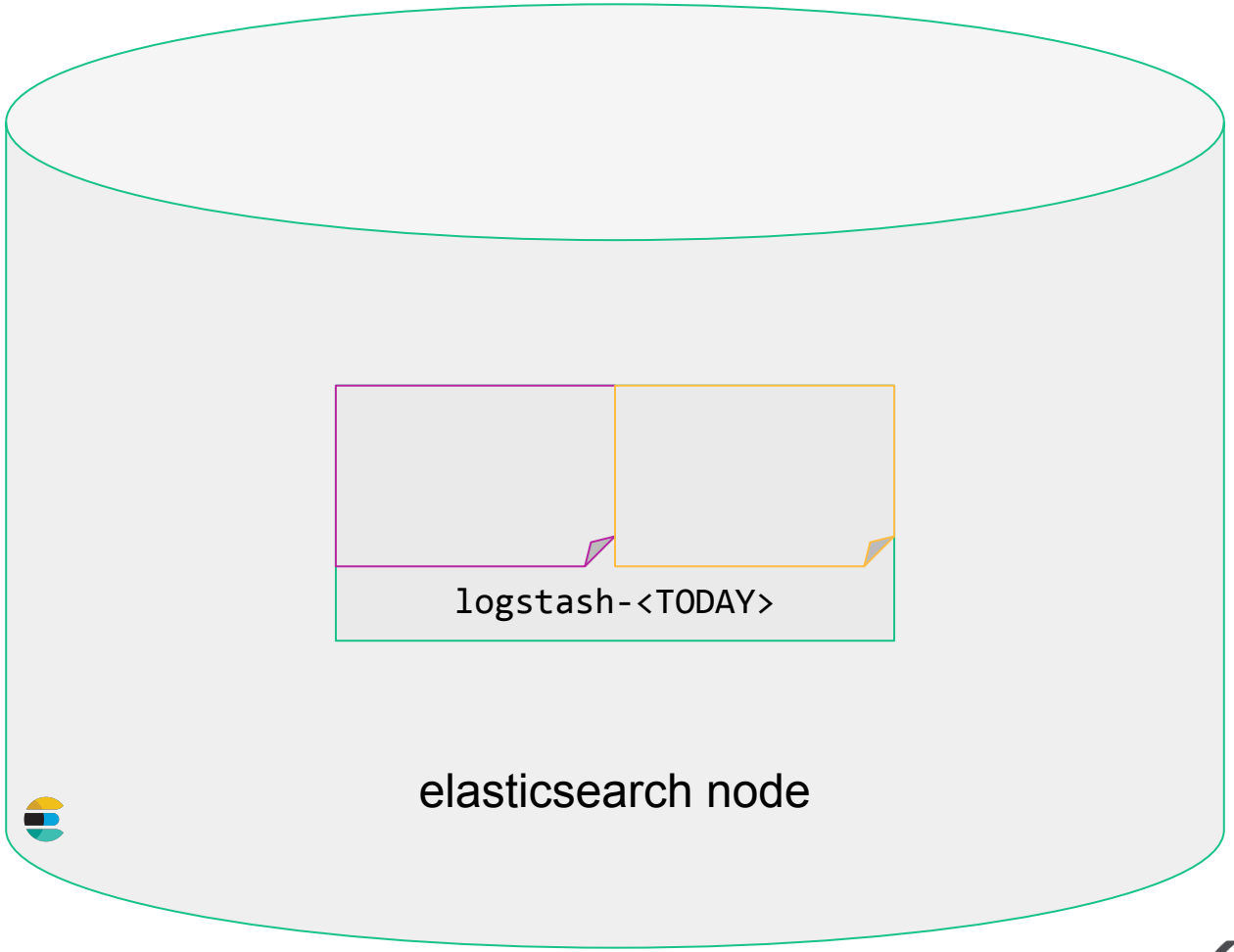
Meltwater



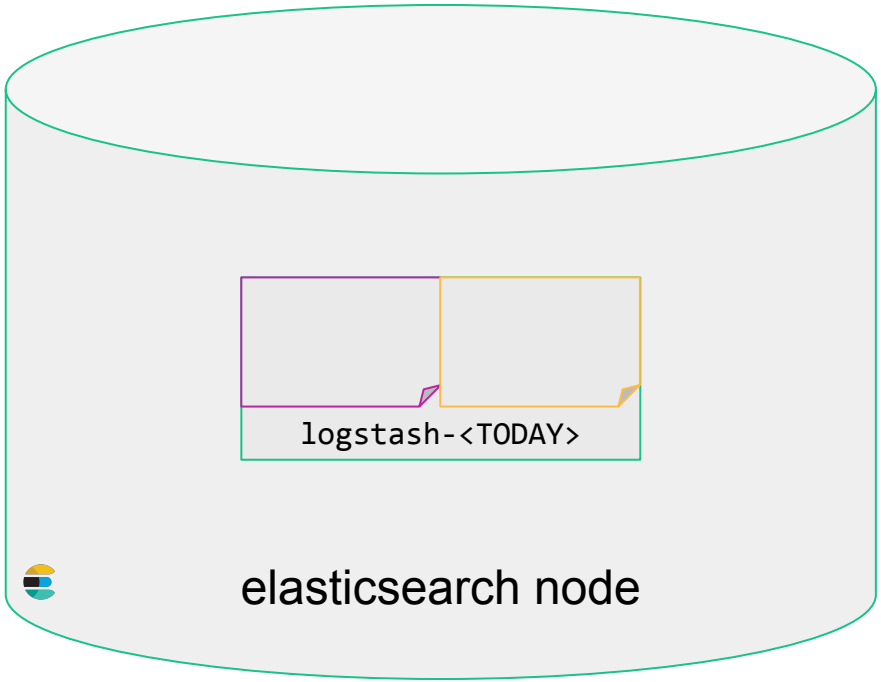
Meltwater

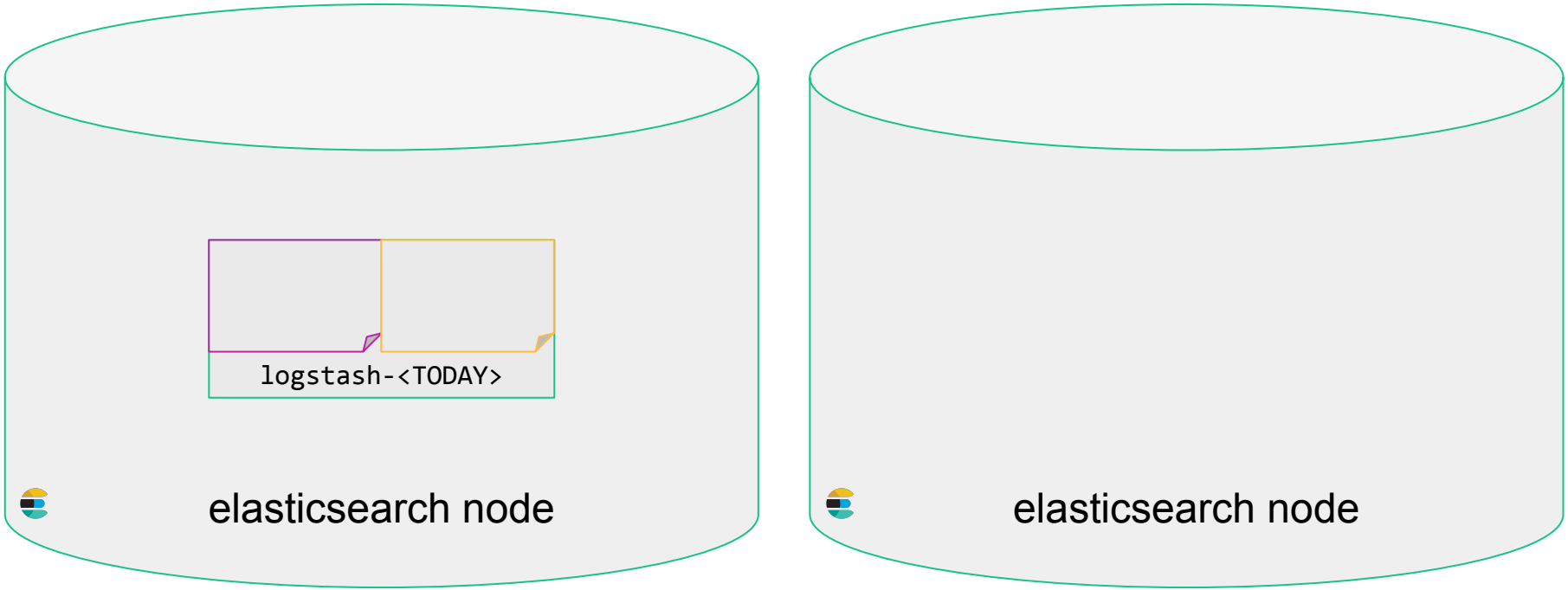

syslog
receiver
app
receiver-ng
redis
indexer
es
kibana

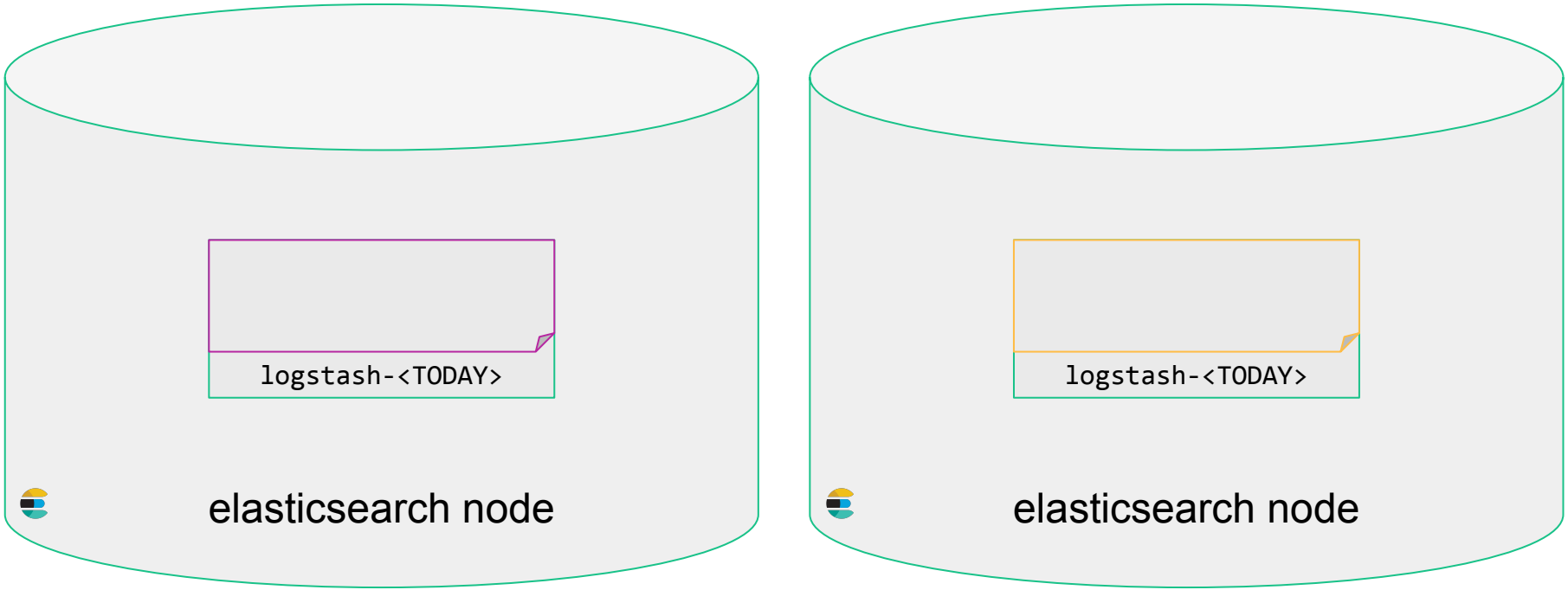


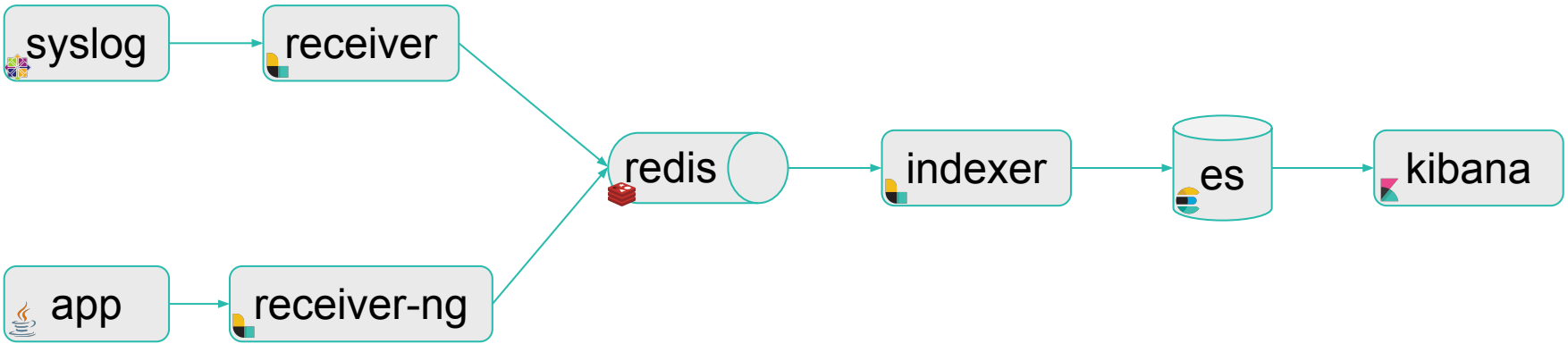


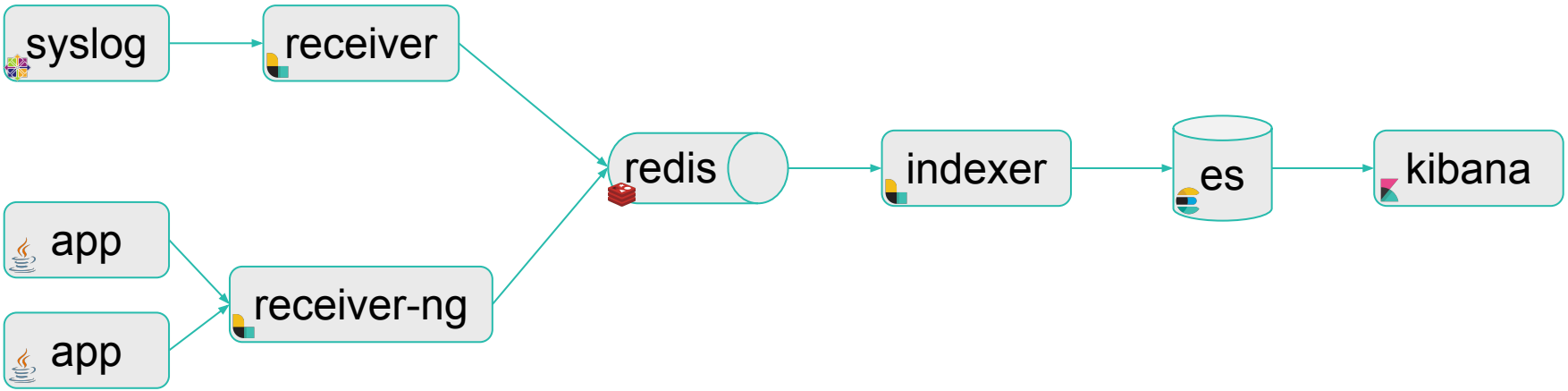
Meltwater

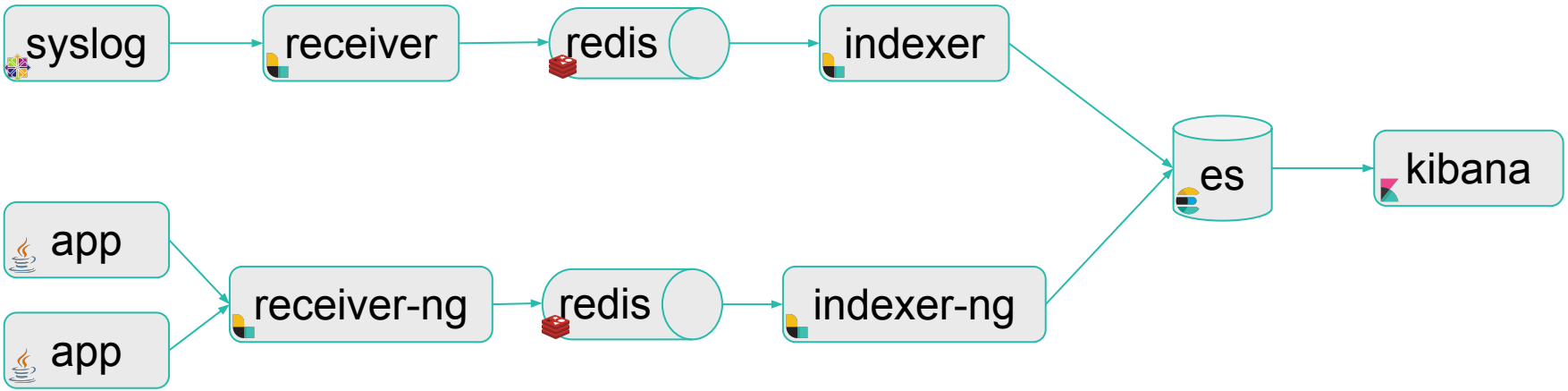

logstash-<TODAY>
elasticsearch node

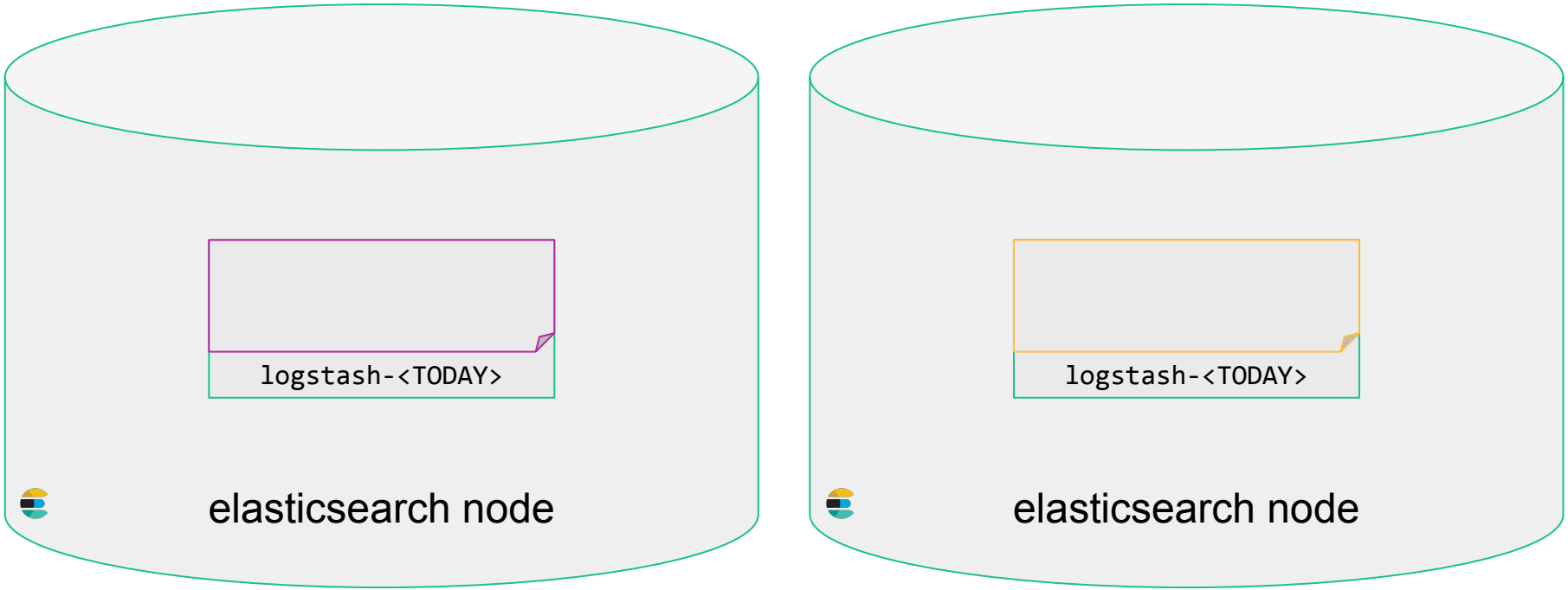


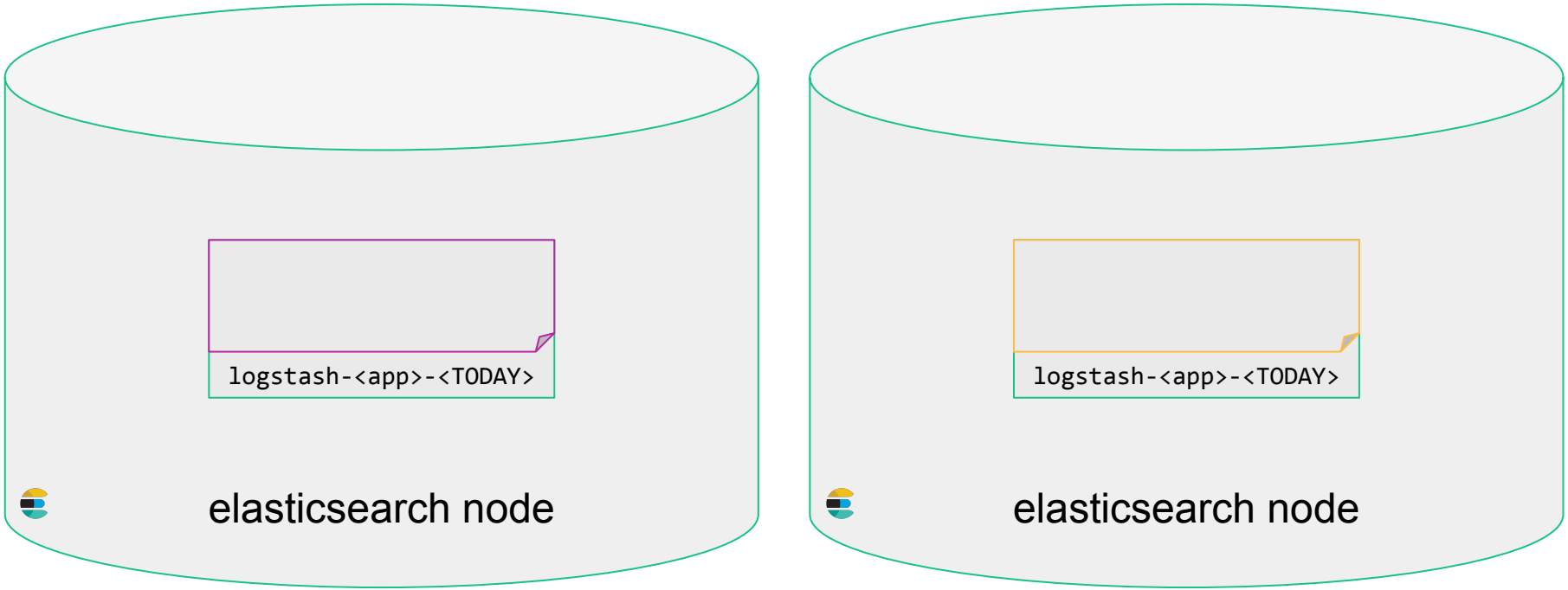

logstash-<TODAY>
elasticsearch node
logstash-<TODAY>
elasticsearch node


syslog
receiver
app
receiver-ng
redis
indexer
es
kibana

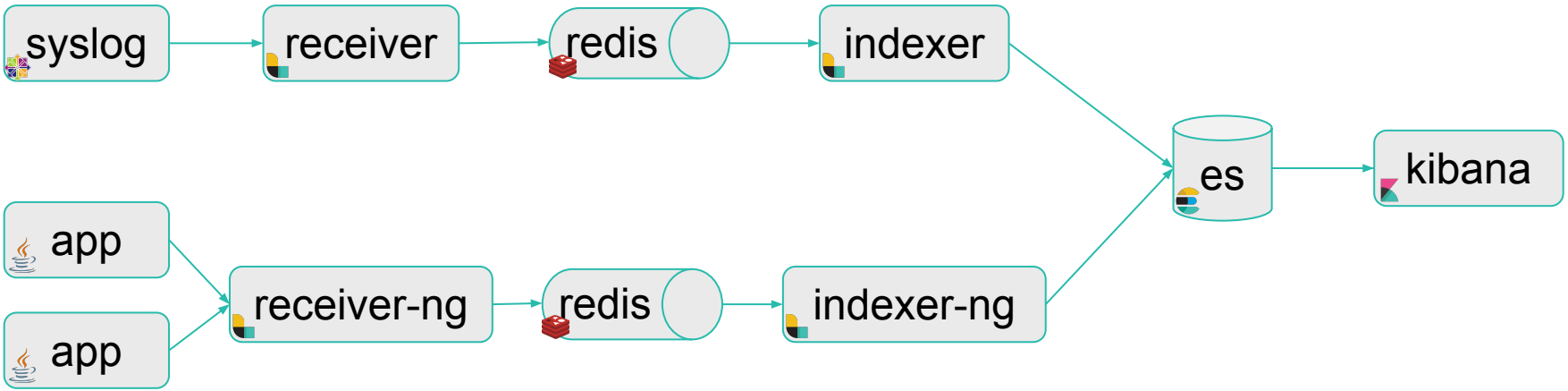

syslog
receiver
app
app
receiver-ng
redis
indexer
es
kibana

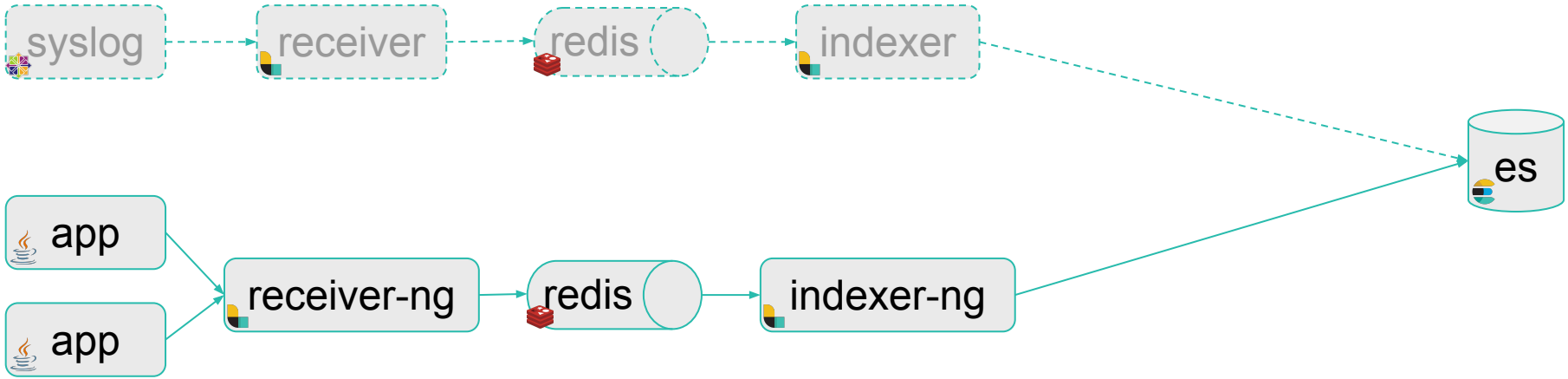

syslog
receiver
redis
indexer
app
app
receiver-ng
redis
indexer-ng
es
kibana


logstash-<TODAY>
elasticsearch node
logstash-<TODAY>
elasticsearch node

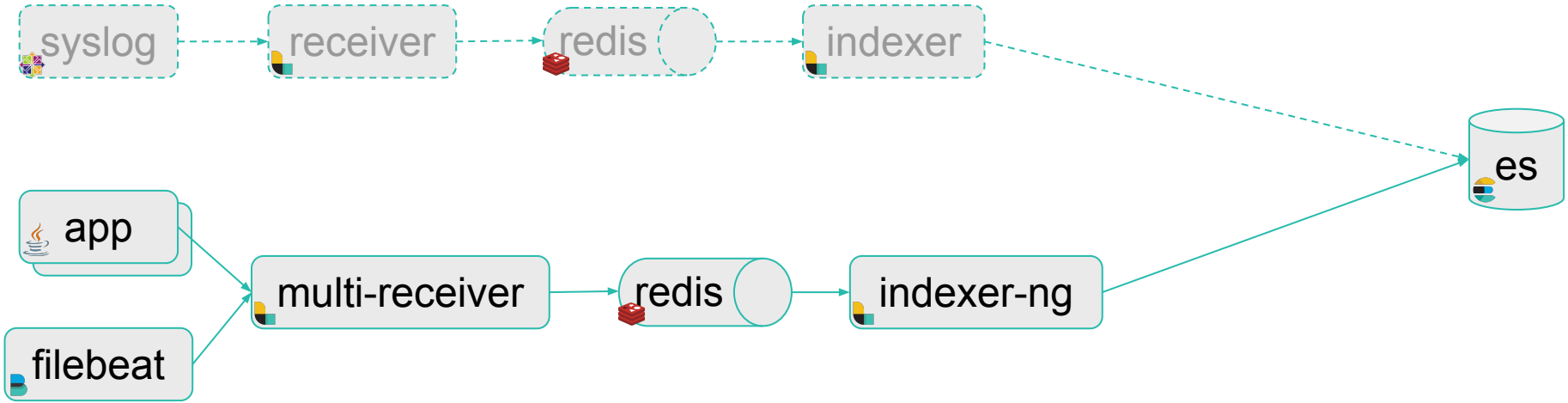

logstash-<app>-<TODAY>
elasticsearch node
logstash-<app>-<TODAY>
elasticsearch node

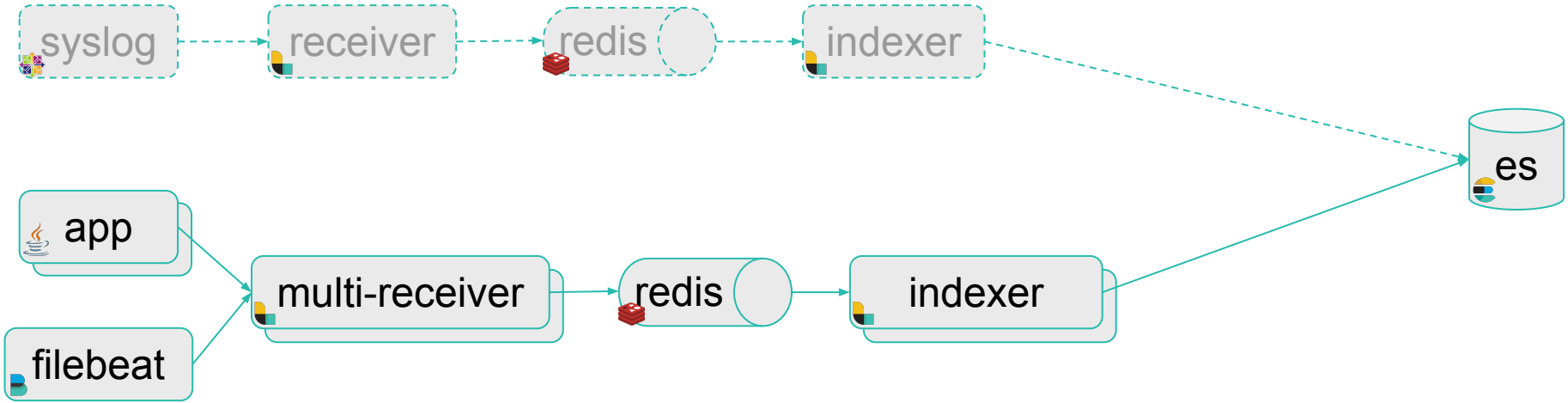
Meltwater


syslog
receiver
redis
indexer
app
app
receiver-ng
redis
indexer-ng
es
kibana


syslog
receiver
redis
indexer
es
app
app
receiver-ng
redis
indexer-ng

Meltwater

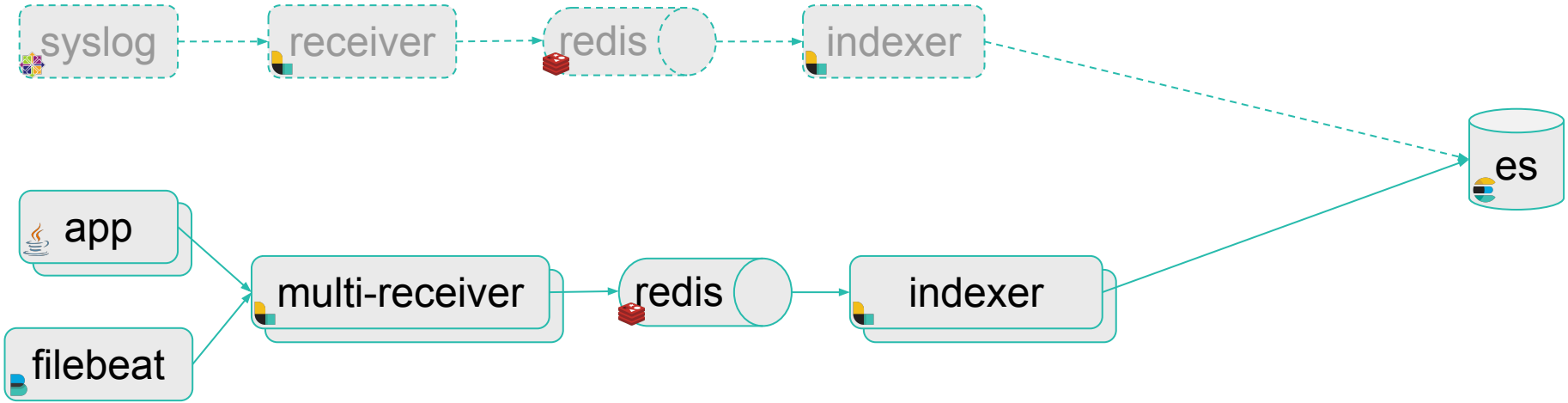

syslog
receiver
redis
indexer
es
app
multi-receiver
redis
indexer-ng
filebeat

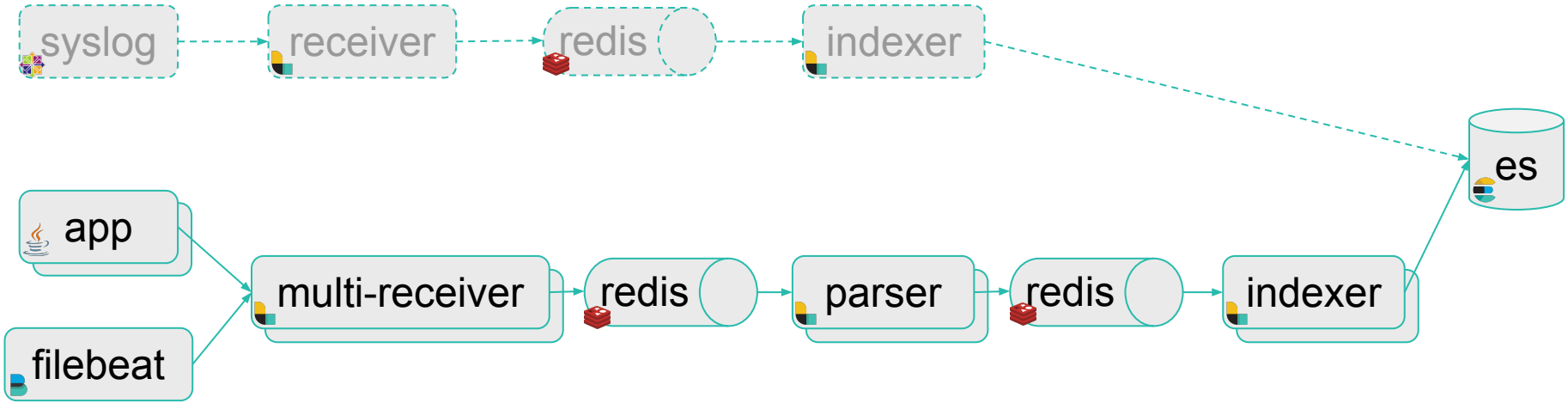

syslog
receiver
redis
indexer
es
app
multi-receiver
redis
indexer
filebeat

```
9.136.49.28 - - [27/Nov/2017:08:29:44 +0100] "GET /dashboard-legacy/v3/maps/dashboard-legacy.js.map HTTP/1.1"
200 1684812 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_1) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/62.0.3202.94 Safari/537.36"
```


```
{
        "ua-name": "Chrome",
        "request": "/dashboard-legacy/v3/maps/dashboard-legacy.js.map",
        "agent": "\"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_1)
        AppleWebKit/537.36 (KHTML, like Gecko) Chrome/62.0.3202.94
        Safari/537.36\"",
        "auth": "-",
        "ua-os_major": "10",
        "ident": "-",
        "ua-os_minor": "13",
        "ua-device": "Other",
        "type": "apache_httpd",
        "ua-patch": "3202",
        "ua-minor": "0",
        "client_location": "Gothenburg",
        "clientip": "79.136.49.28",
        "ua-os_name": "Mac OS X",
        "ua-os": "Mac OS X 10.13.1",
        "geoip": {
                "timezone": "Europe/Stockholm",
                "ip": "79.136.49.28",
                "latitude": 57.7167,
                "continent_code": "EU",
                "city_name": "Gothenburg",
                "country_name": "Sweden",
                "country_code2": "SE",
                "country_code3": "SE",
                "region_name": "Västra Götaland",
                "location": {
                        "lon": 11.9667,
                        "lat": 57.7167
                },
                "postal_code": "400 16",
                "region_code": "O",
                "longitude": 11.9667
        }
}
```


Meltwater


syslog
receiver
redis
indexer
es
app
multi-receiver
redis
indexer
filebeat


syslog
receiver
redis
indexer
es
app
multi-receiver
redis
parser
redis
indexer
filebeat

Bug

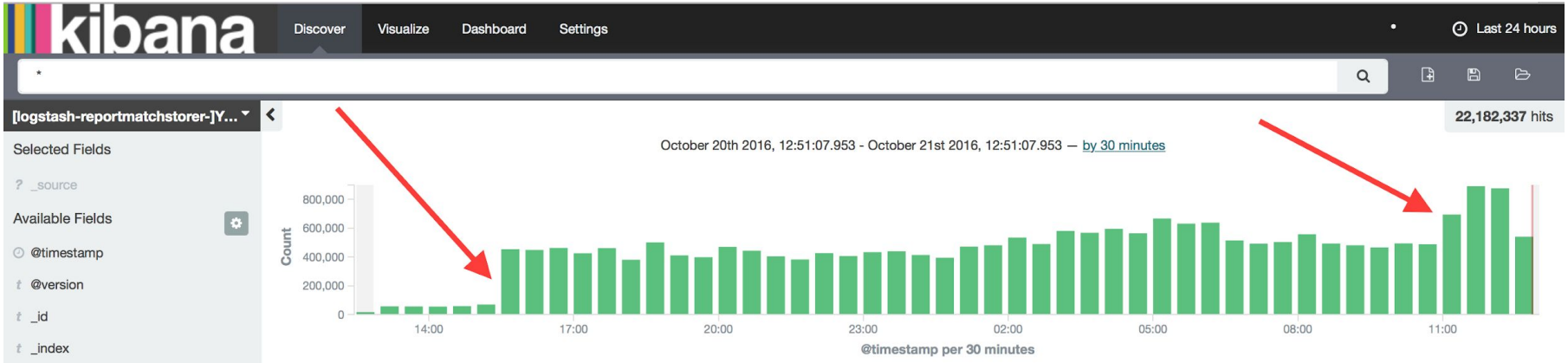
json_lines codec

```
{"message": "Stuff happened"}\n
{"message": "Stuff happened"}\n
{"message": "Stuff happened"}\n
{"message": "Stuff happened"}\n
EOF
```

json codec

```
{"message": "Stuff happened"}\n
EOF
```

Meltwater


kibana
Discover
Visualize
Dashboard
Settings
Last 24 hours
*
[logstash-reportmatchstorer-]Y...
22,182,337 hits
Selected Fields
? _source
Available Fields
@timestamp
@version
_id
_index
October 20th 2016, 12:51:07.953 - October 21st 2016, 12:51:07.953 — by 30 minutes
Count
800,000
600,000
400,000
200,000
0
14:00
17:00
20:00
23:00
02:00
05:00
08:00
11:00
@timestamp per 30 minutes

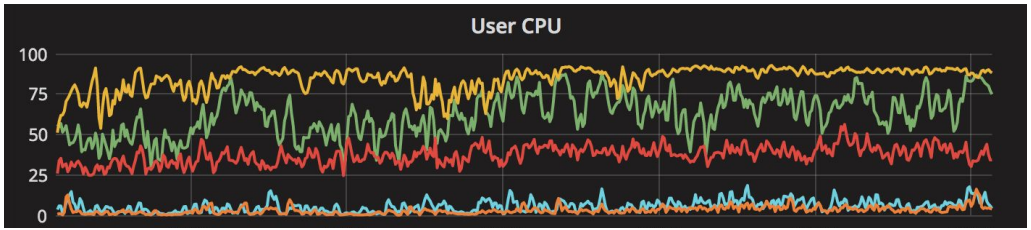
Meltwater

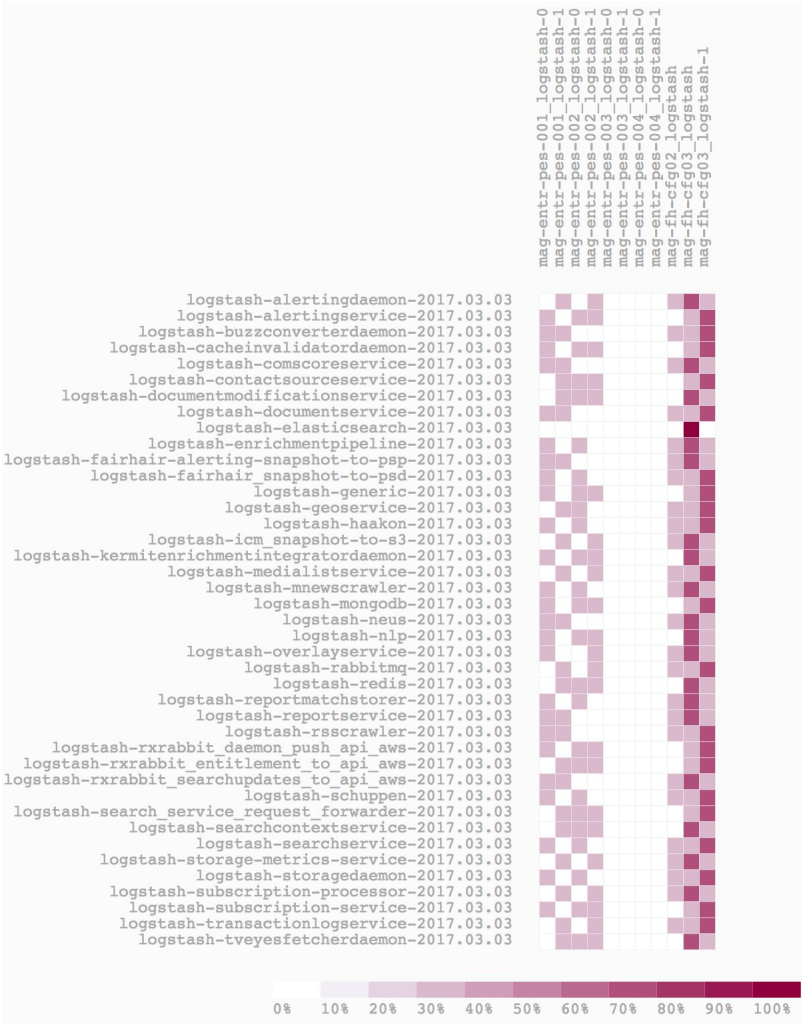
Challenge #1

Meltwater

Challenge

- Massive elasticsearch scaling issues
- Capacity problems (no more)
- Uneven cluster
  - Running on spare hw
  - Allocation catch-22





Challenge

- Massive elasticsearch scaling issues
- Capacity problems (no more)
- Uneven cluster
  - Running on spare hw
  - Allocation catch-22

Solution

- Build new cluster (migrate to AWS)
  - Use opportunity to upgrade stack

Easy-peasy!

Challenge #2

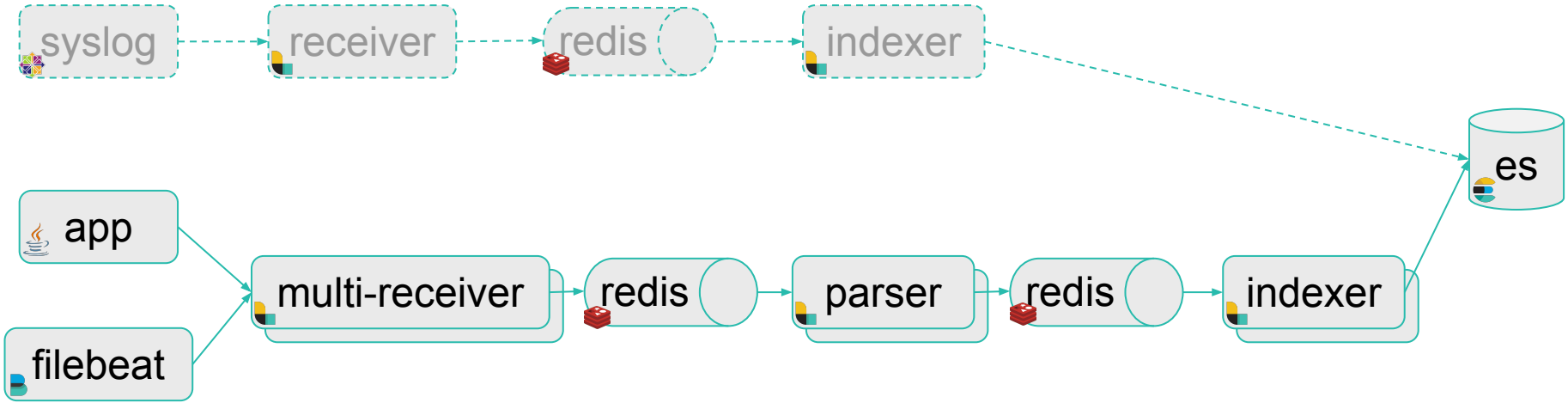
Meltwater

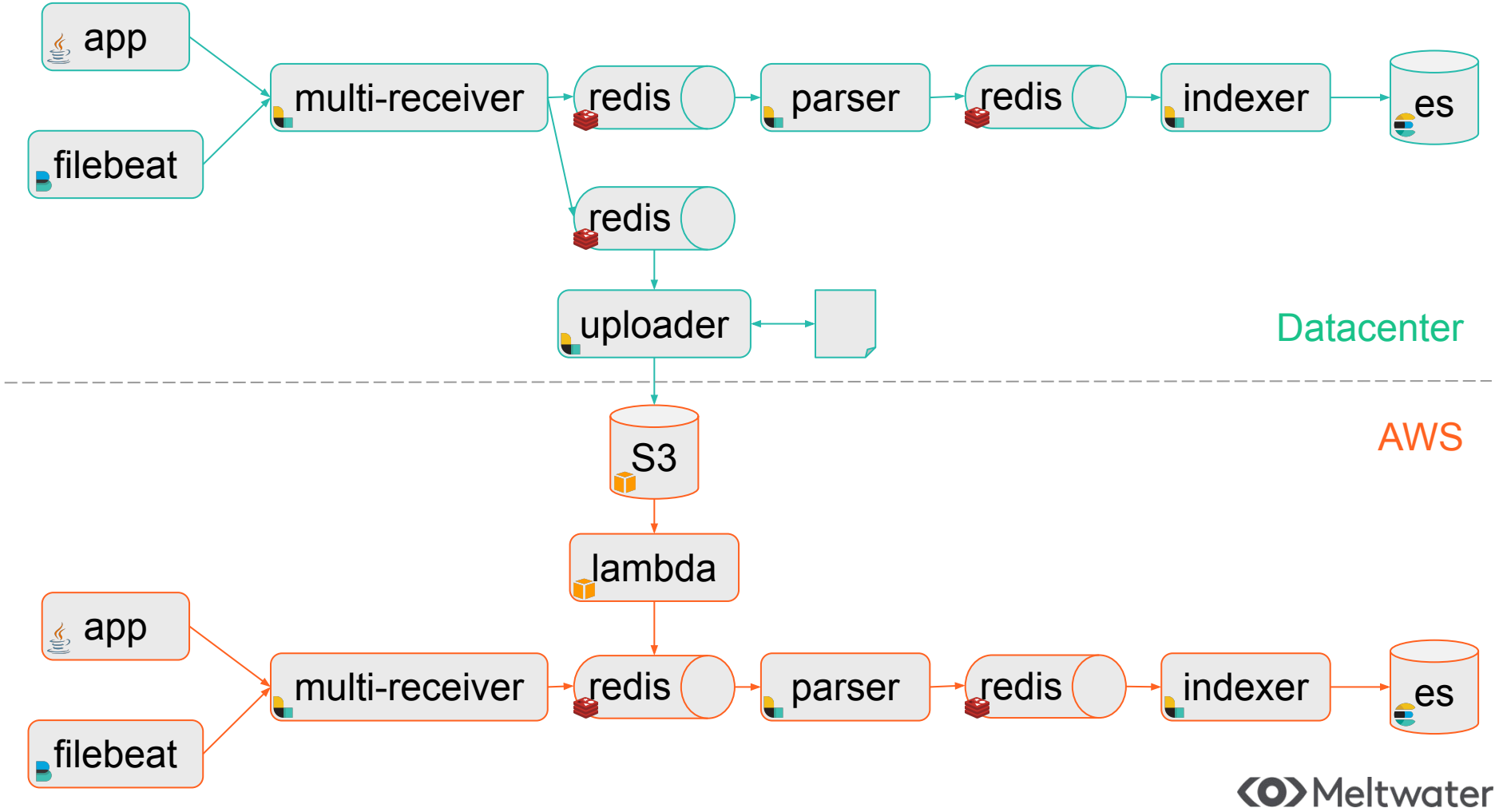
Challenge

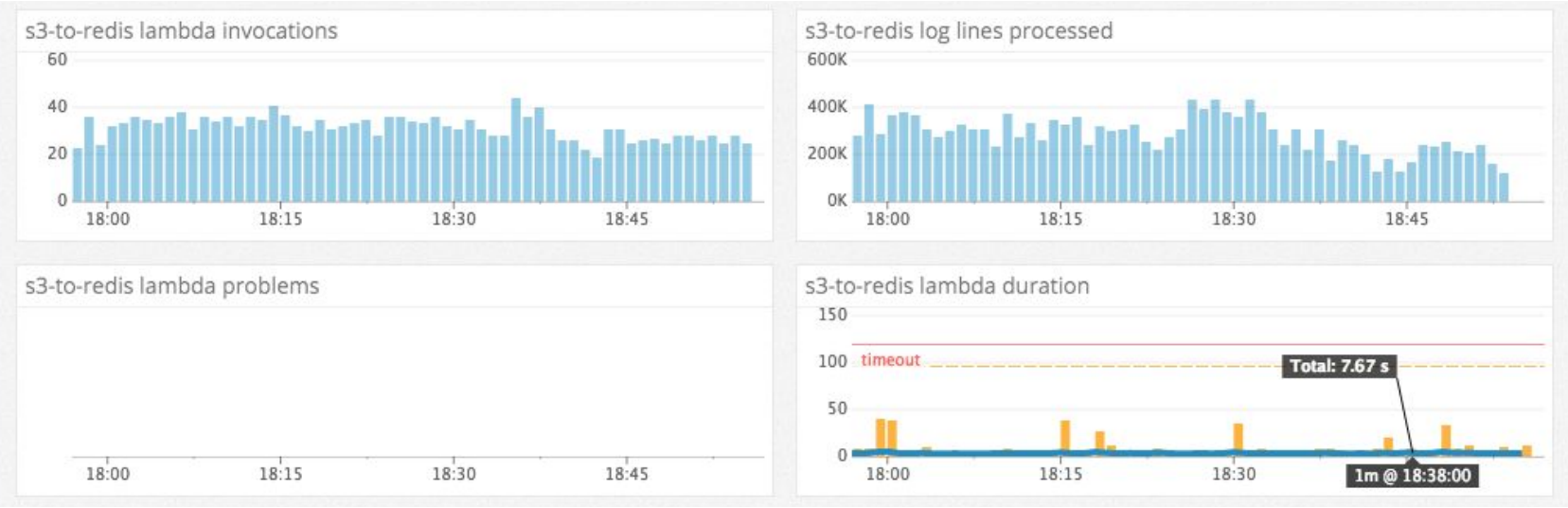
- Migrate to AWS...

Solution

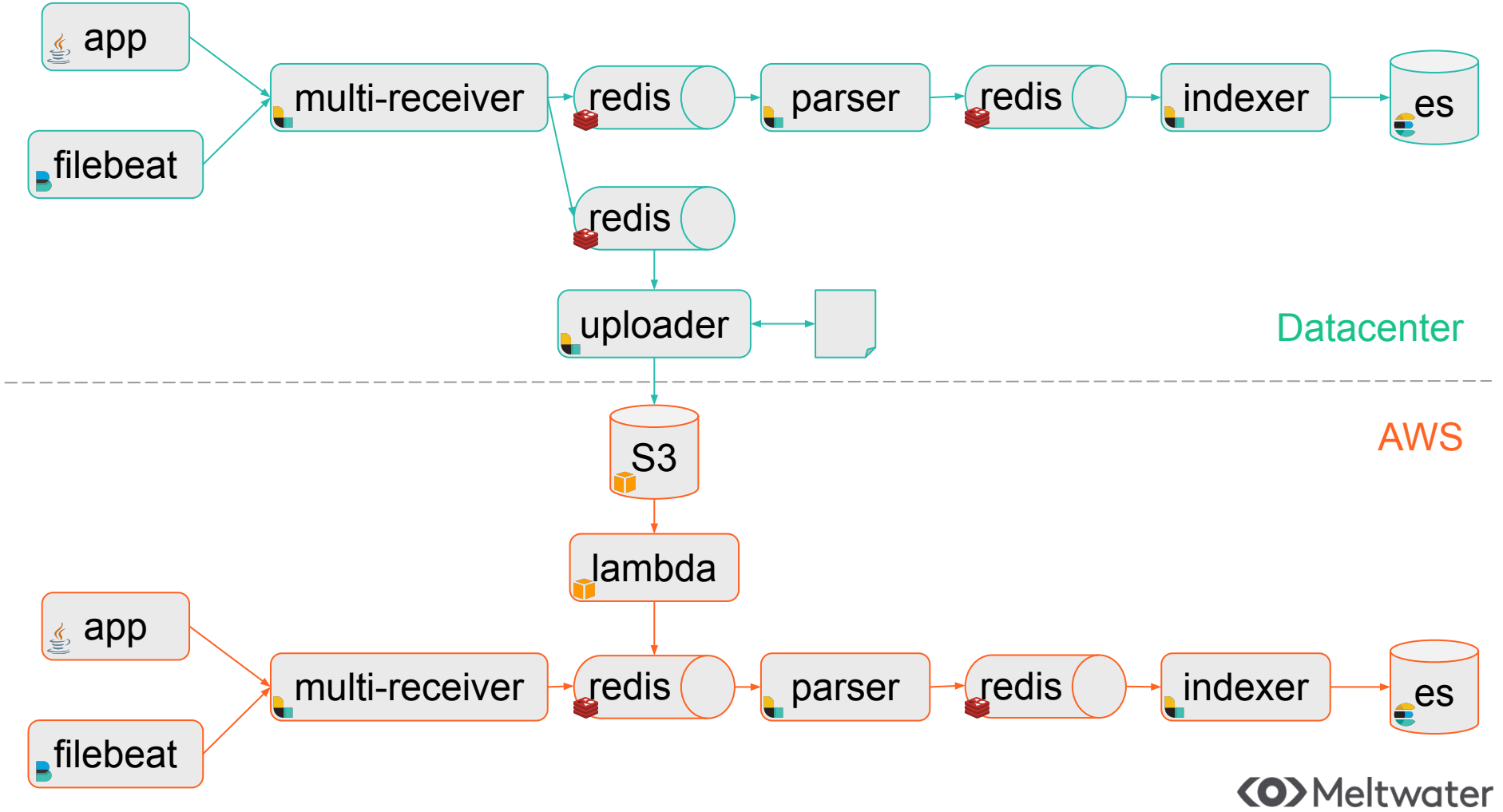
- Build new mesos cluster
- Build new and shiny elasticsearch cluster
- Send data to both places

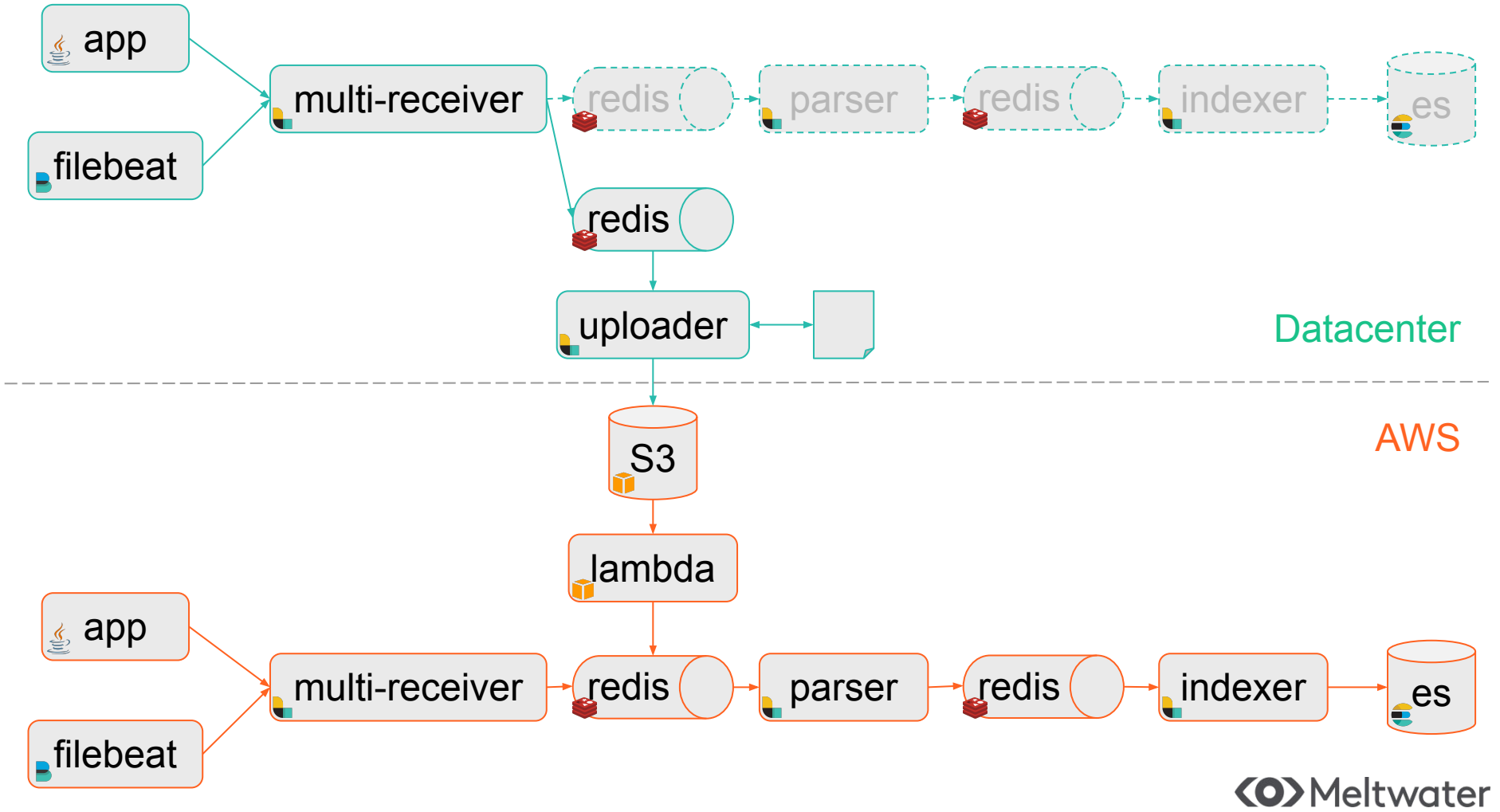

syslog
receiver
redis
indexer
es
app
multi-receiver
redis
parser
redis
indexer
filebeat


app
filebeat
multi-receiver
redis
parser
redis
indexer
es
redis
uploader
Datacenter
AWS
S3
lambda
app
filebeat
multi-receiver
redis
parser
redis
indexer
es
Meltwater


s3-to-redis lambda invocations
60
40
20
0
18:00
18:15
18:30
18:45
s3-to-redis log lines processed
600K
400K
200K
0K
18:00
18:15
18:30
18:45
s3-to-redis lambda problems
18:00
18:15
18:30
18:45
s3-to-redis lambda duration
150
100
50
0
timeout
Total: 7.67 s
18:00
18:15
18:30
1m @ 18:38:00

Meltwater


app
filebeat
multi-receiver
redis
parser
redis
indexer
es
redis
uploader
Datacenter
AWS
S3
lambda
app
filebeat
multi-receiver
redis
parser
redis
indexer
es
Meltwater

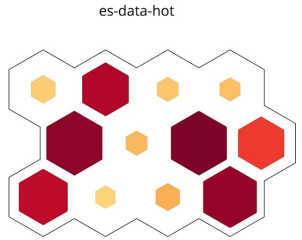

app
filebeat
multi-receiver
redis
parser
redis
indexer
es
redis
uploader
Datacenter
AWS
S3
lambda
app
filebeat
multi-receiver
redis
parser
redis
indexer
es
Meltwater

Challenge #3

Meltwater

Challenge

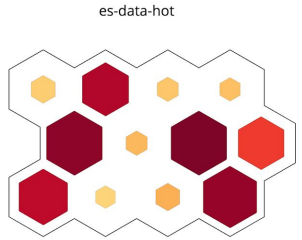
- Elasticsearch hotspots

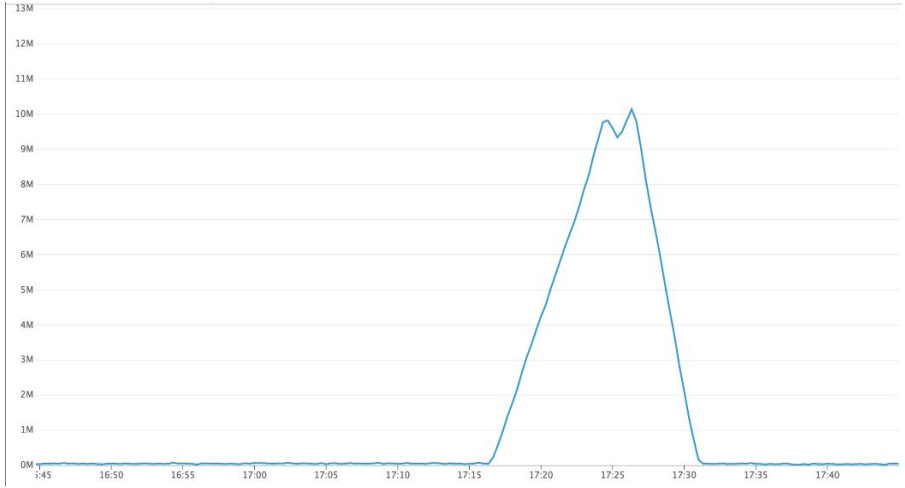


Meltwater

Challenge

- Elasticsearch hotspots
- Hotspot causes an indexing backlog

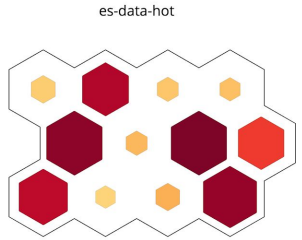


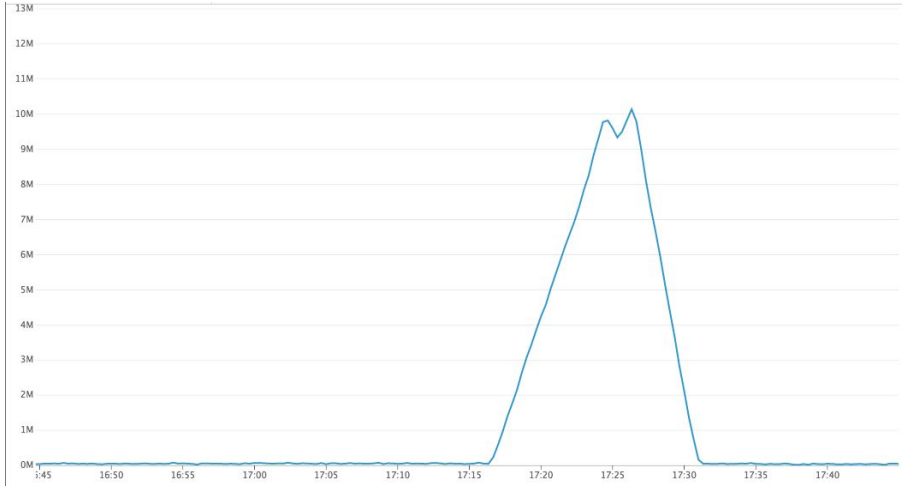


Meltwater

Challenge

- Elasticsearch hotspots
- Hotspot causes an indexing backlog
- Potential root causes
  - Uneven distribution of indices to nodes
  - Data surge





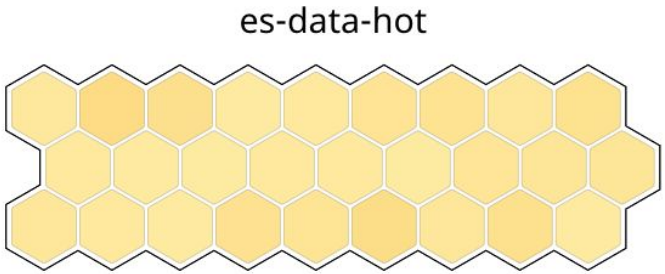
Meltwater

Challenge

- Elasticsearch hotspots
- Hotspot causes an indexing backlog
- Potential root causes
  - Uneven distribution of indices to nodes
  - Data surge

Solution

- ~~Manually solve distribution problem~~
  - ~~Move indices around~~
- Don’t run the system at 85% capacity
  - Achieve better distribution



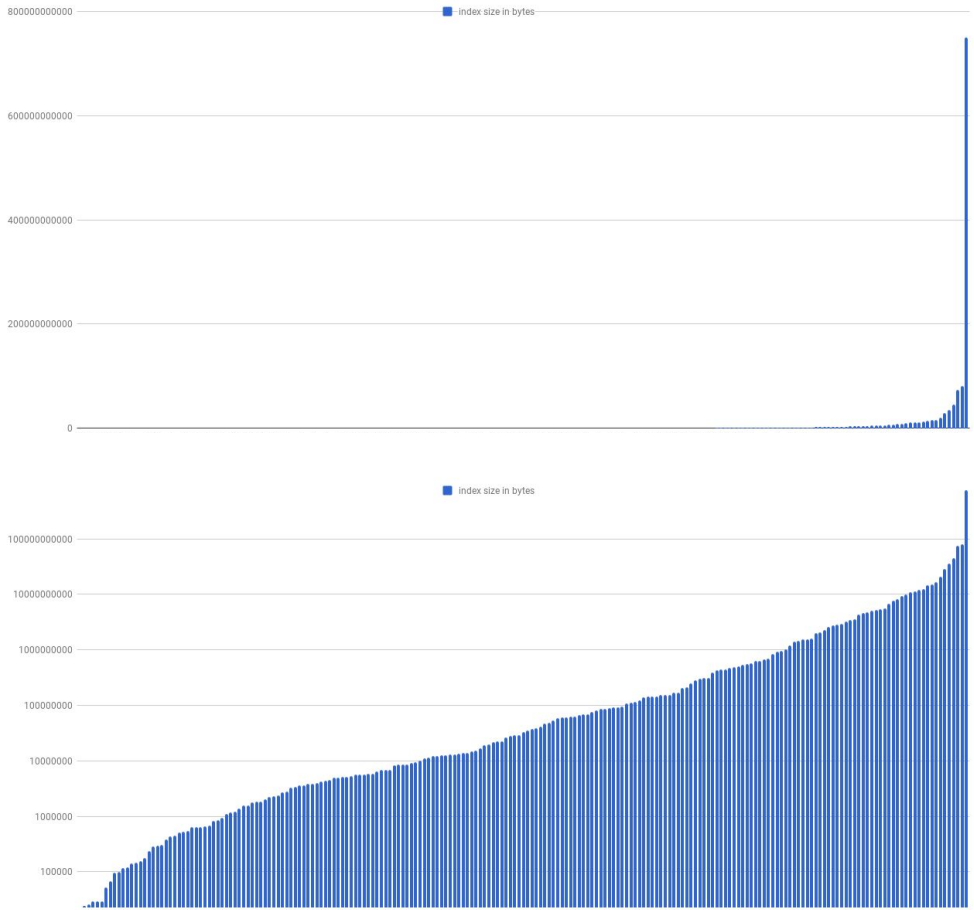
Meltwater

Challenge #4

Meltwater

Challenge

- Elasticsearch hotspots
- Hotspot causes an indexing backlog
- Potential root causes
  - Uneven distribution of index sizes
  - Too much data per index shard
  - Number of shards per index hardcoded



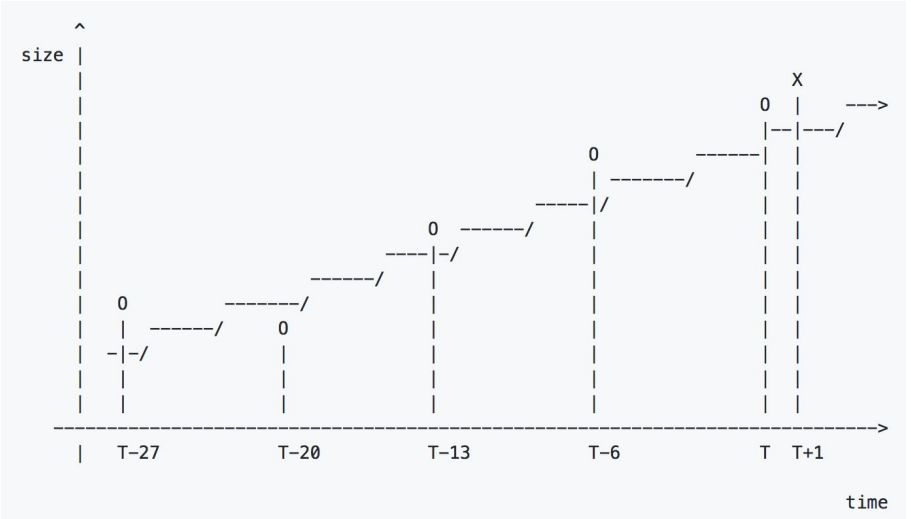
Meltwater

Challenge

- Elasticsearch hotspots
- Hotspot causes an indexing backlog
- Potential root causes
  - Uneven distribution of index sizes
  - Too much data per index shard
  - Number of shards per index hardcoded

Solution

- Try to right-size indices
  - Appropriate amount of shards
  - Based on historical data



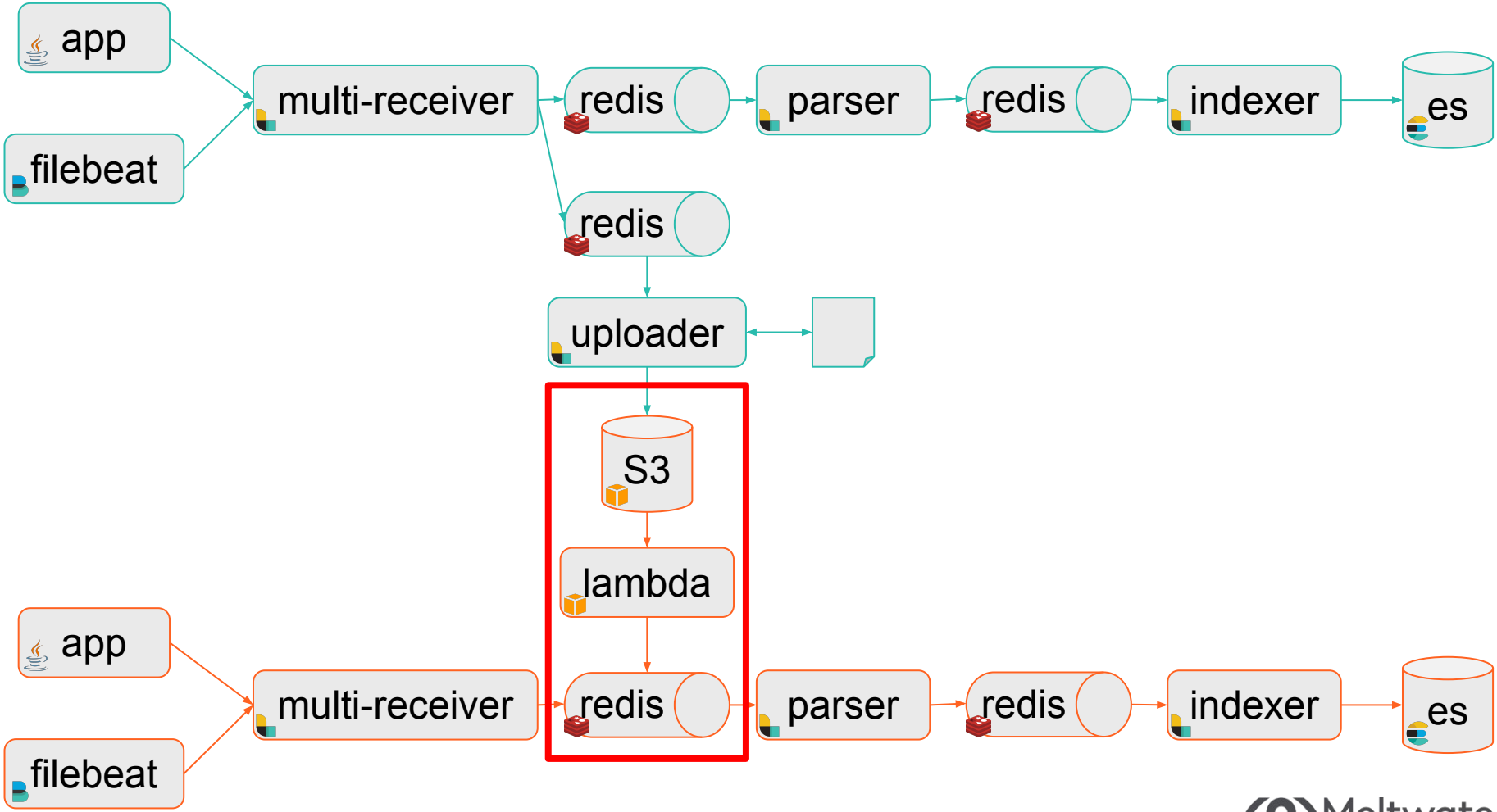
Meltwater

Challenge #5

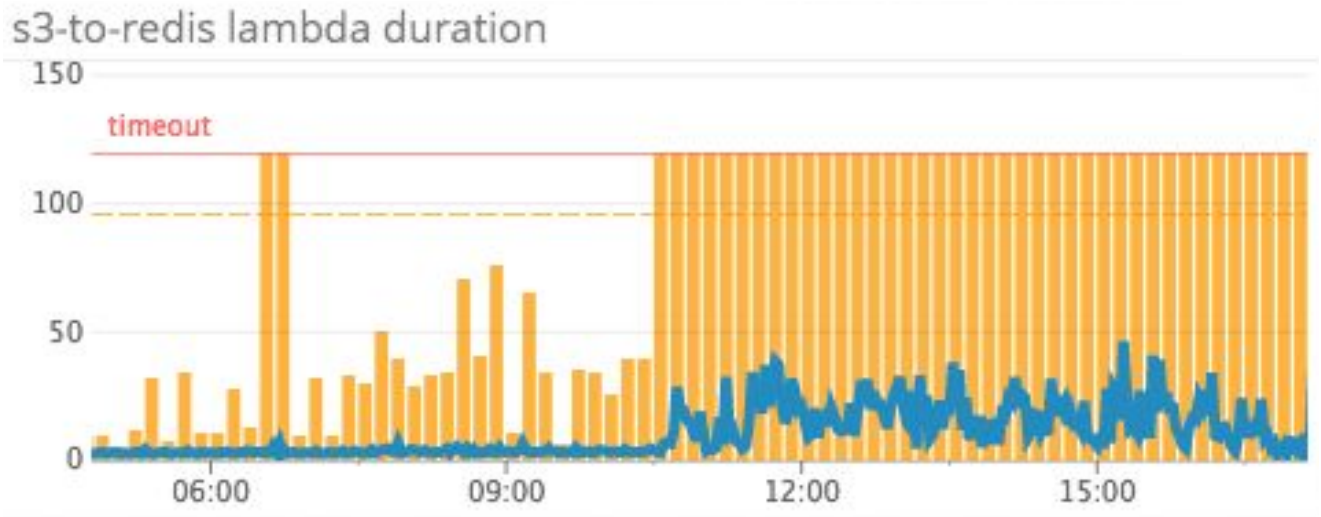
Meltwater

Challenge

- Redis not appropriate anymore
  - In-memory
  - Single-threaded
  - Not a queue (data duplication)
  - Not distributed
  - Hard to scale


app
filebeat
multi-receiver
redis
parser
redis
indexer
es
redis
uploader
S3
lambda
app
filebeat
multi-receiver
redis
parser
redis
indexer
es

Meltwater


s3-to-redis lambda duration
150
timeout
100
50
0
06:00
09:00
12:00
15:00

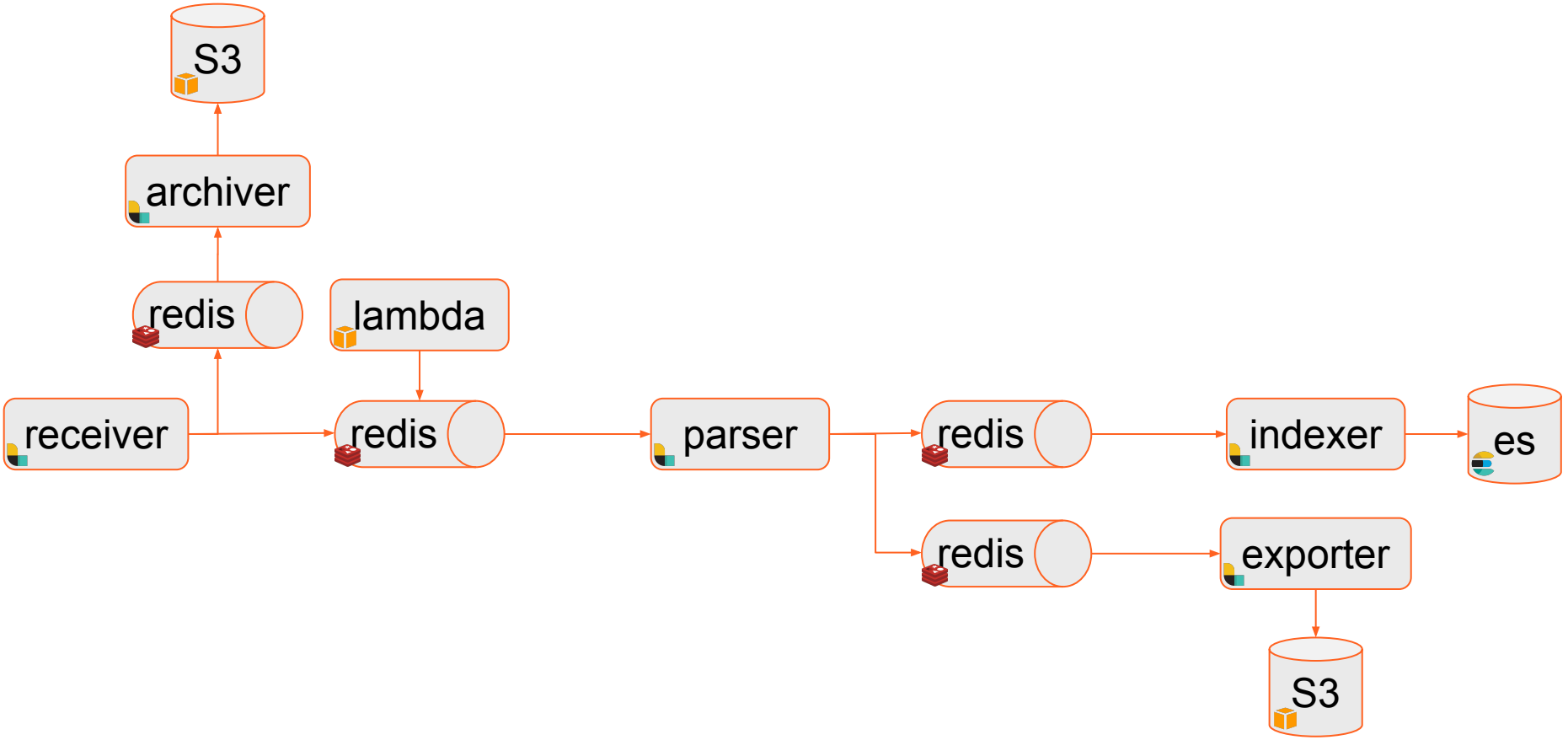
Meltwater

Challenge

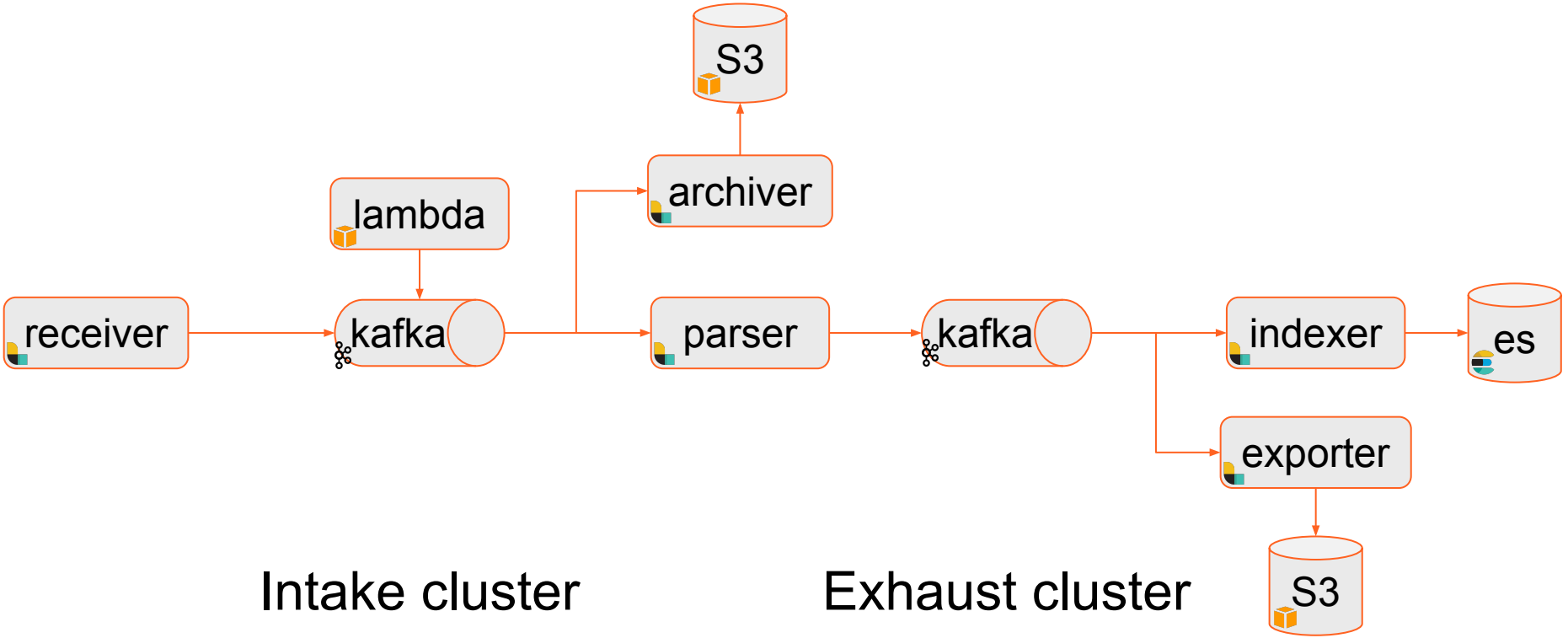
- Redis not appropriate anymore
  - In-memory
  - Single-threaded
  - Not a queue (data duplication)
  - Not distributed
  - Hard to scale

Solution

- Replace Redis with Kafka
  - Disk-based
  - Multi-threaded
  - Append-only log
  - Distributed
  - Horizontally scalable


S3
archiver
redis
lambda
receiver
redis
parser
redis
indexer
es
redis
exporter
S3

Meltwater


S3
lambda
archiver
receiver
kafka
parser
kafka
indexer
es
exporter
S3
Intake cluster
Exhaust cluster

Intake cluster

- Main focus: fast and safe
  - Data not backed up yet
  - Consumers CPU-focused

Exhaust cluster

- Main focus: huge buffer
  - All data can be replayed
  - Consumers I/O-focused

Intake cluster

- Main focus: fast and safe
  - Data not backed up yet
  - Consumers CPU-focused
- More nodes

Exhaust cluster

- Main focus: huge buffer
  - All data can be replayed
  - Consumers I/O-focused
- Fewer nodes

Intake cluster

- Main focus: fast and safe
  - Data not backed up yet
  - Consumers CPU-focused
- More nodes
- Small, fast disks

Exhaust cluster

- Main focus: huge buffer
  - All data can be replayed
  - Consumers I/O-focused
- Fewer nodes
- Huge, slower disks

Intake cluster

- Main focus: fast and safe
  - Data not backed up yet
  - Consumers CPU-focused
- More nodes
- Small, fast disks
- Higher replication

Exhaust cluster

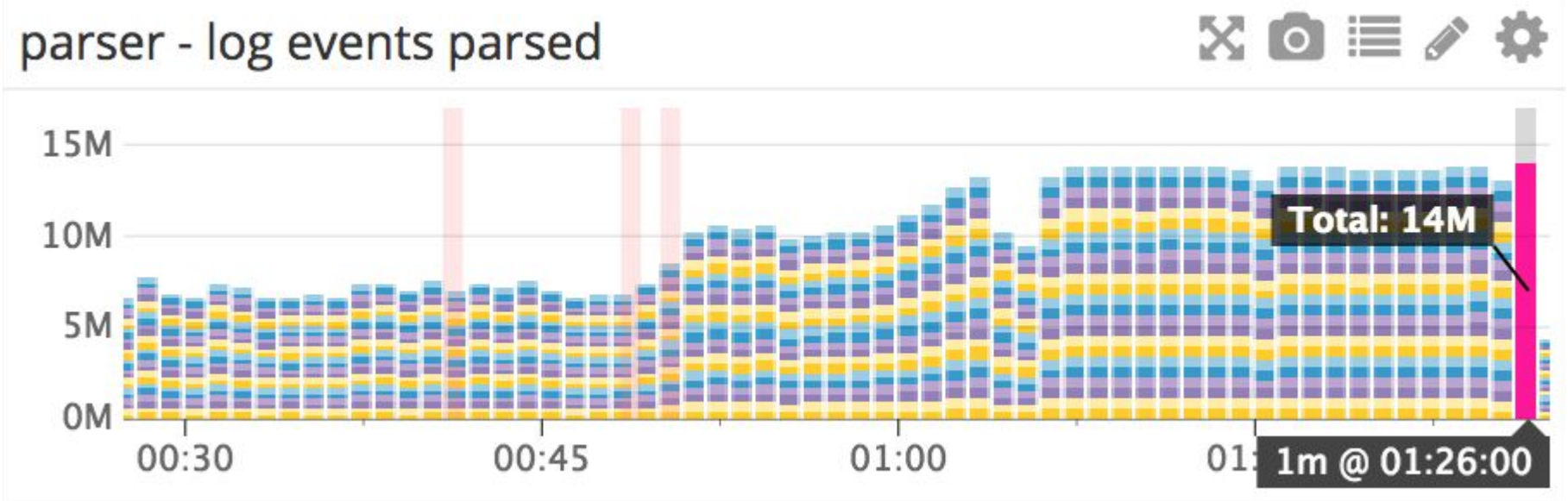
- Main focus: huge buffer
  - All data can be replayed
  - Consumers I/O-focused
- Fewer nodes
- Huge, slower disks
- Lower replication

Intake cluster

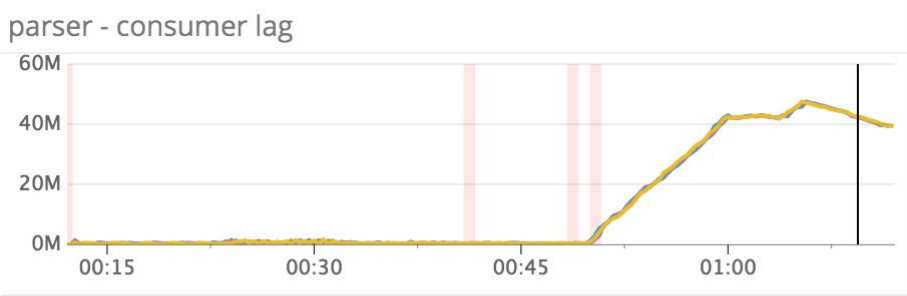
- EC2 type: i3.xlarge
  - 4 CPUs
  - 30GB RAM
- Disk: NVMe SSD
  - 0.95 TB
- Count: 3
- Kafka replication: 3
- Partitions per topic: 48

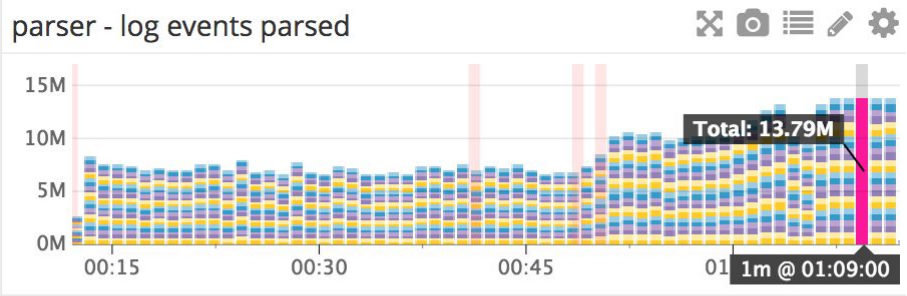
Exhaust cluster

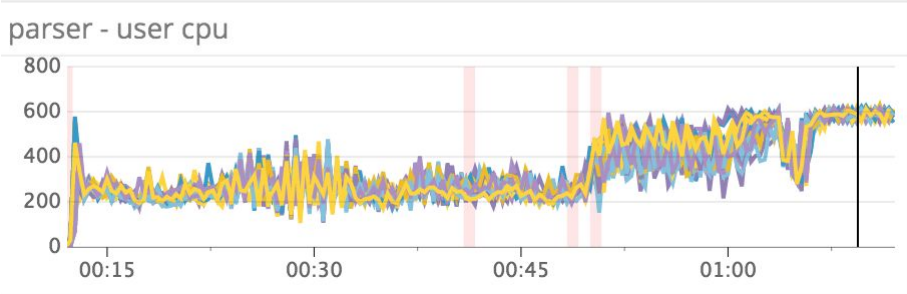
- EC2 type: r4.xlarge
  - 4 CPUs
  - 30GB RAM
- Disk: st1 EBS
  - 6 TB
- Count: 2
- Kafka replication: 2
- Partitions per topic: 48

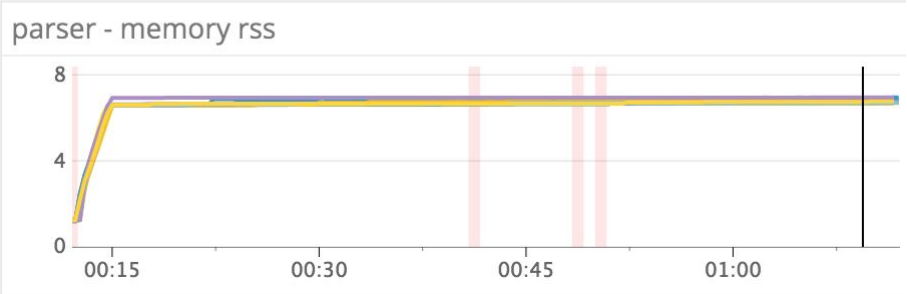

parser - log events parsed
15M
10M
5M
0M
00:30
00:45
01:00
Total: 14M
1m @ 01:26:00

Meltwater


parser - consumer lag
60M
40M
20M
0M
00:15
00:30
00:45
01:00


parser - log events parsed
15M
10M
5M
0M
Total: 13.79M
00:15
00:30
00:45
01
1m @ 01:09:00


parser - user cpu
800
600
400
200
0
00:15
00:30
00:45
01:00


parser - memory rss
8
4
0
00:15
00:30
00:45
01:00

Meltwater

14M / min = 233 333 / s

Still bottlenecked on consumer cpu

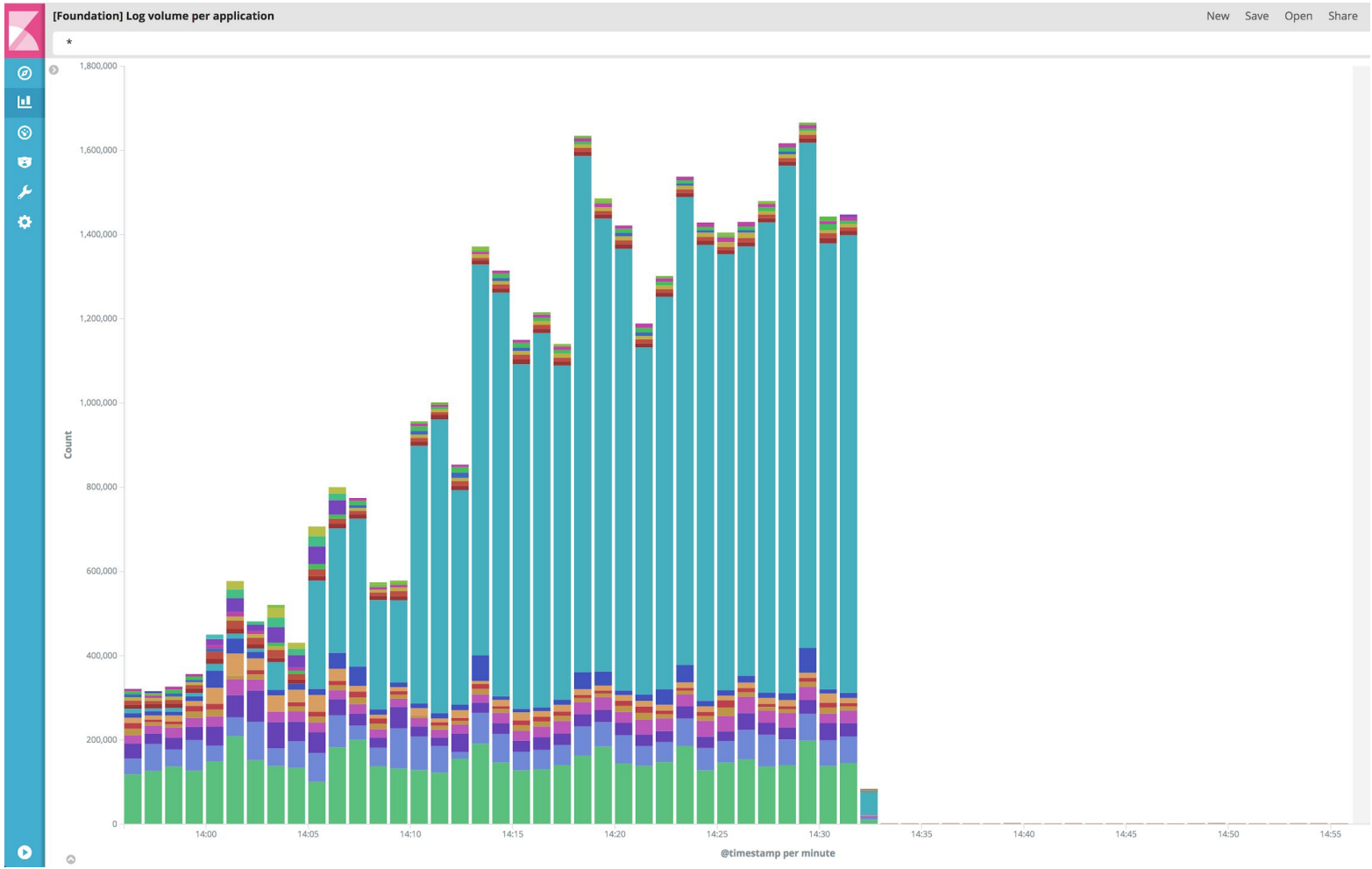
Kafka as is can do much more

And can still scale

What next?

Meltwater

```
-       <root level="INFO">
+       <root level="TRACE">
                <appender-ref ref="LOGSTASH"/>
        </root>
```


[Foundation] Log volume per application
New
Save
Open
Share
*
1,800,000
1,600,000
1,400,000
1,200,000
1,000,000
800,000
600,000
400,000
200,000
0
Count
14:00
14:05
14:10
14:15
14:20
14:25
14:30
14:35
14:40
14:45
14:50
14:55
@timestamp per minute

Meltwater

Challenge

- Hard to react to data spikes
  - Indexing falls behind

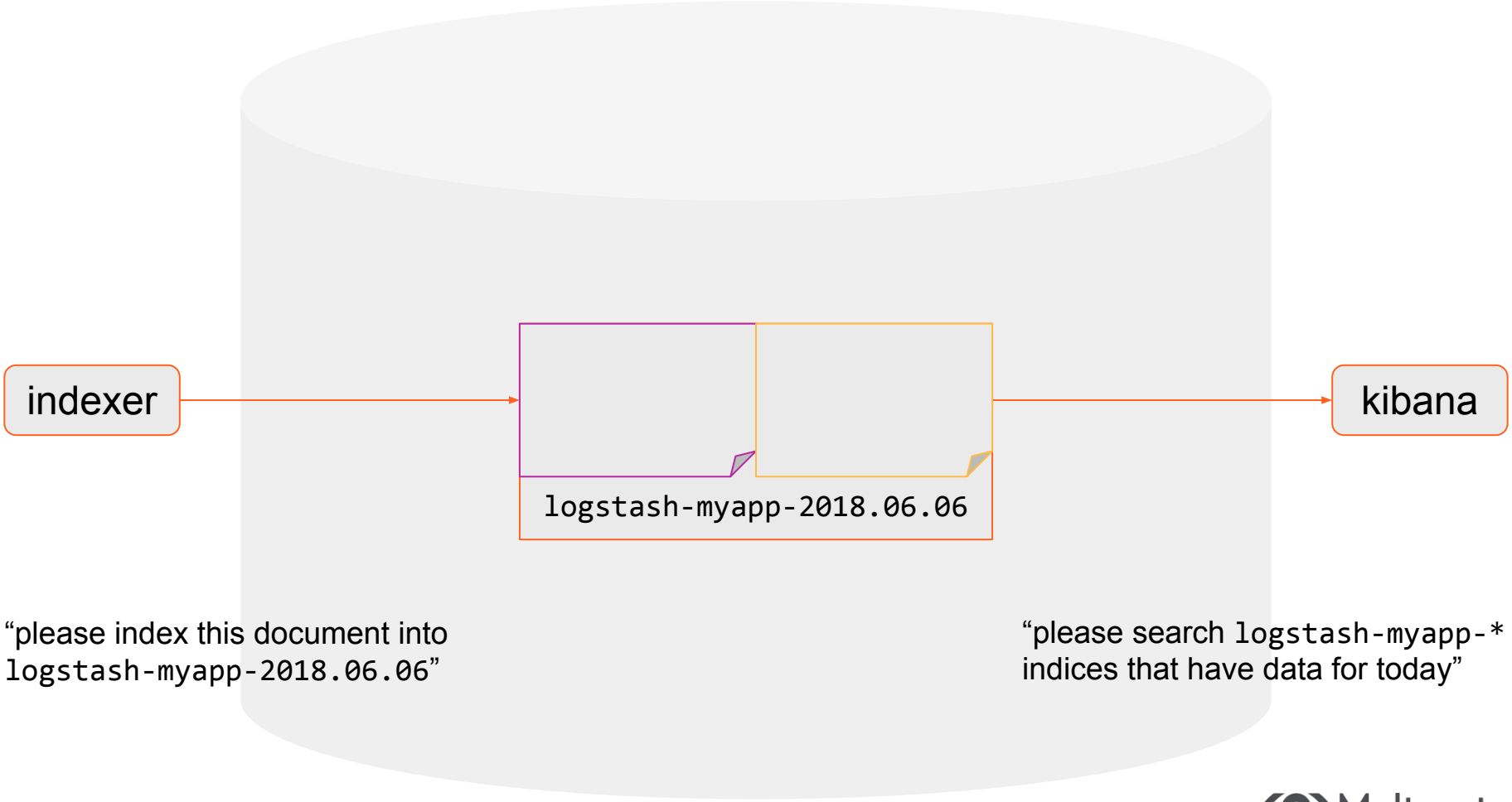
Solution

- Use Elasticsearch index aliases
- Use Elasticsearch rollover API

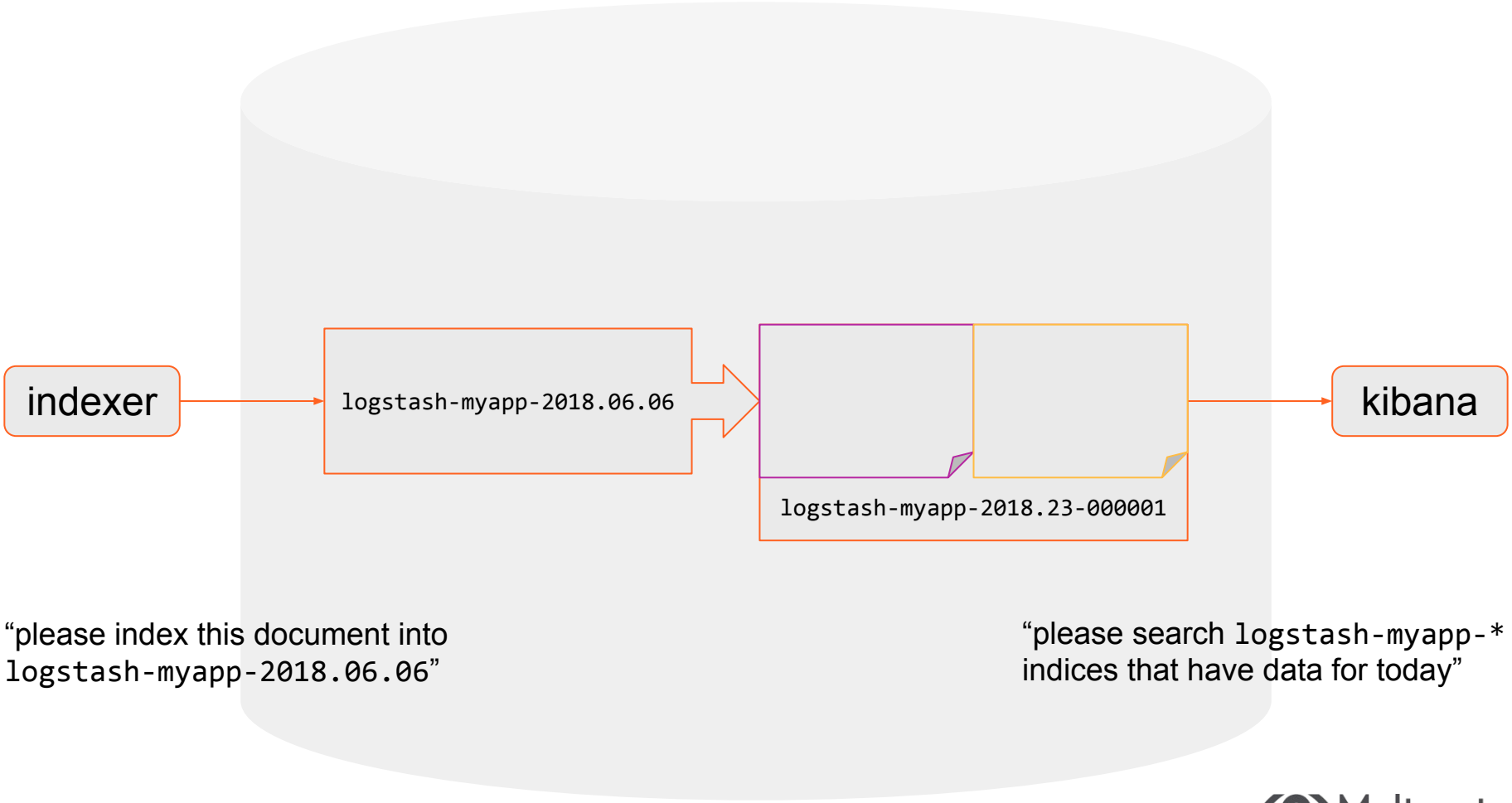
Meltwater


indexer
es
kibana


indexer
kibana


indexer
logstash-myapp-2018.06.06
kibana
“please index this document into logstash-myapp-2018.06.06”
“please search logstash-myapp-* indices that have data for today”

Meltwater

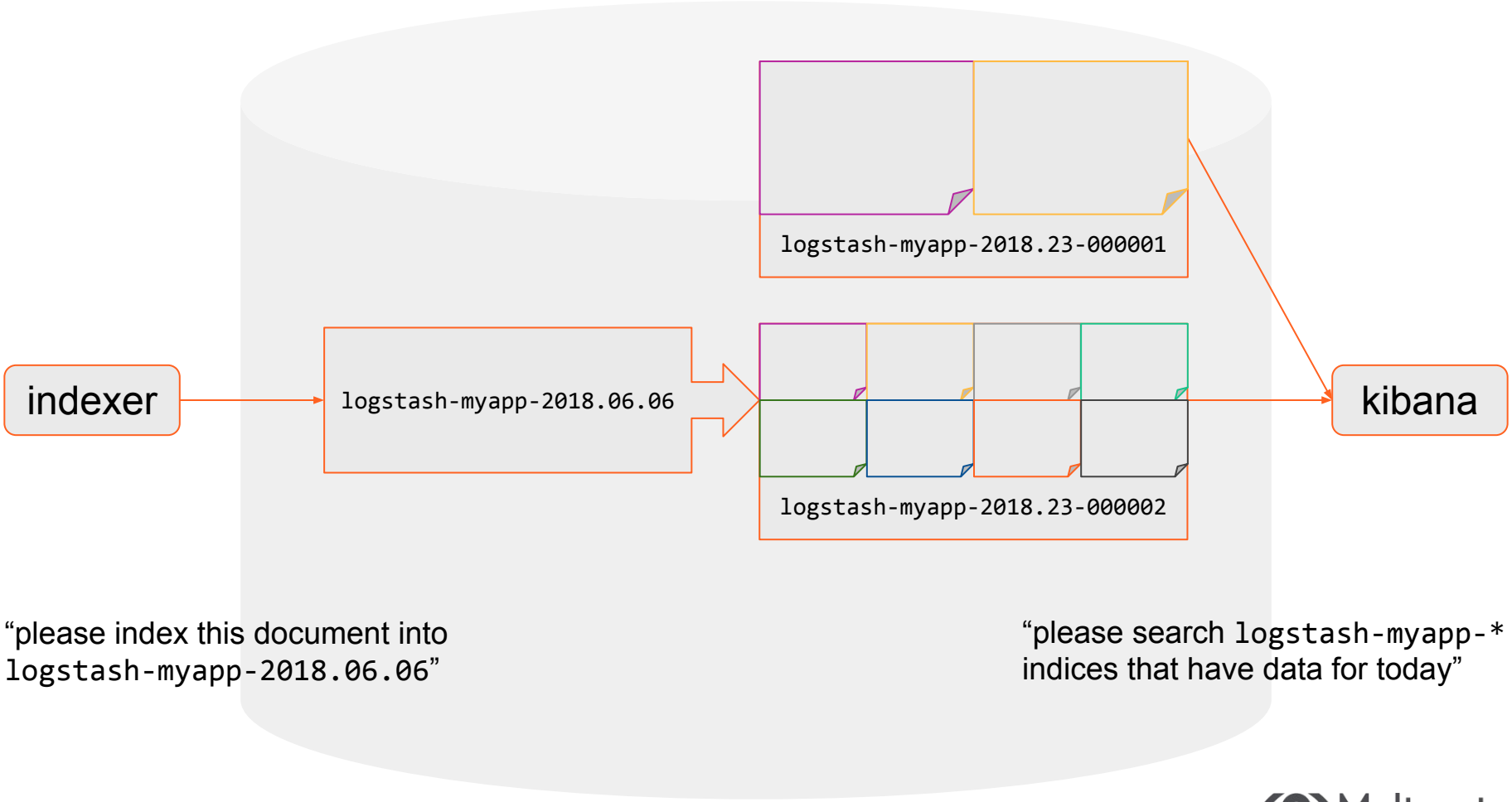

indexer
logstash-myapp-2018.06.06
logstash-myapp-2018.23-000001
kibana
“please index this document into logstash-myapp-2018.06.06”
“please search logstash-myapp-* indices that have data for today”

Meltwater

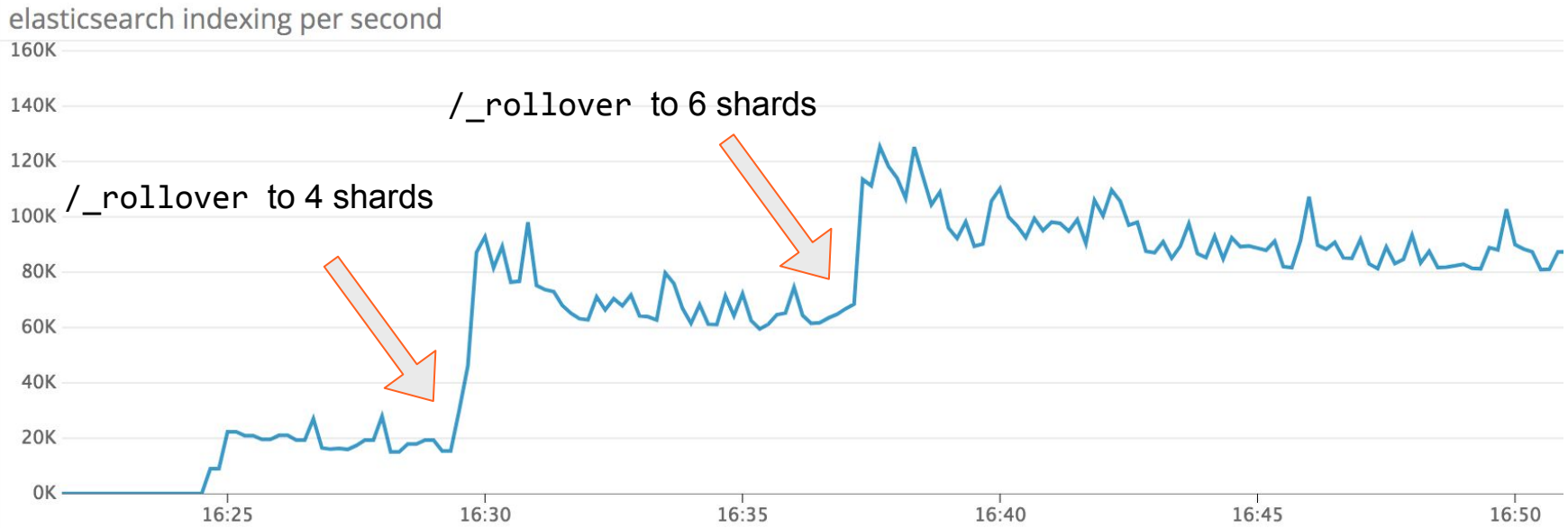
POST /logstash-myapp-2018.06.06/_rollover

```
{
    "settings": {
        "index.number_of_shards": 8
    }
}
```

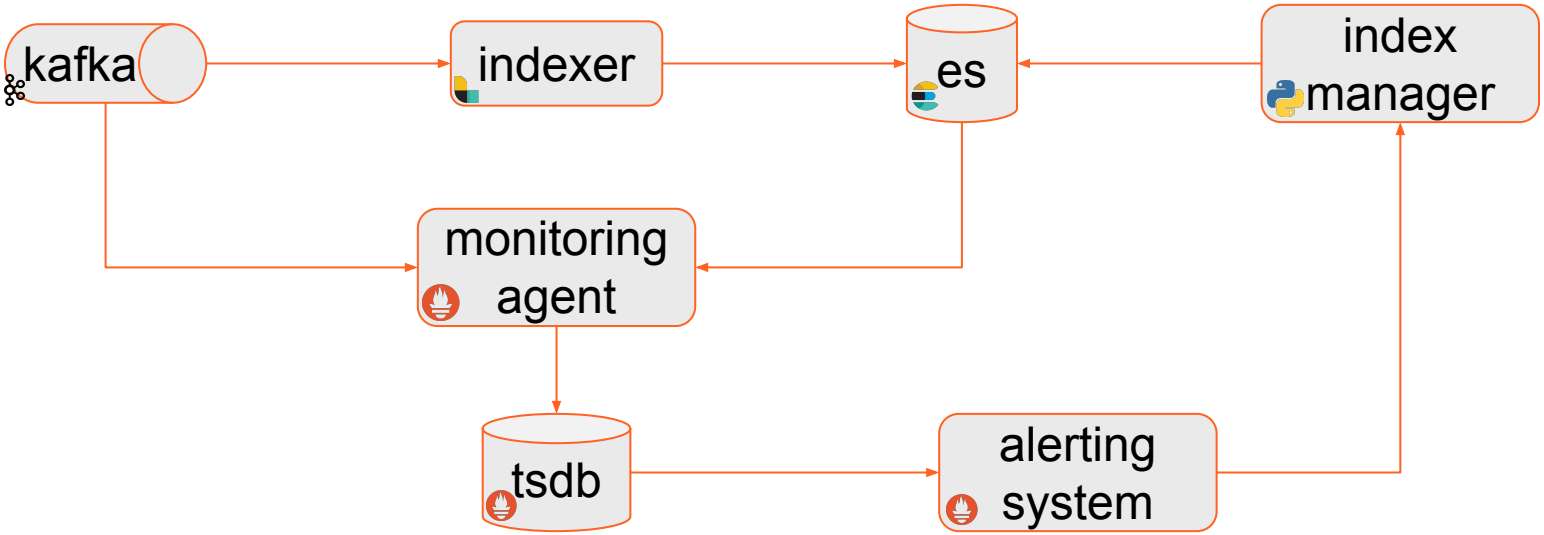
Meltwater


logstash-myapp-2018.23-000001
indexer
logstash-myapp-2018.06.06
logstash-myapp-2018.23-000002
kibana
“please index this document into logstash-myapp-2018.06.06”
“please search logstash-myapp-* indices that have data for today”

Meltwater


elasticsearch indexing per second
160K
140K
120K
100K
80K
60K
40K
20K
0K
16:25
16:30
16:35
16:40
16:45
16:50
/_rollover to 4 shards
/_rollover to 6 shards

Meltwater


kafka
indexer
es
index manager
monitoring agent
tsdb
alerting system

Meltwater

To a billion messages

and (well) beyond

Thanks!

János Csorvási

janos.csorvasi@meltwater.com

linkedin.com/in/jcsorvasi

Meltwater

underthehood.meltwater.com

Meltwater