

RAFAŁ BAJEK

EFFECTIVE LOG MANAGEMENT WITH ELASTICSEARCH

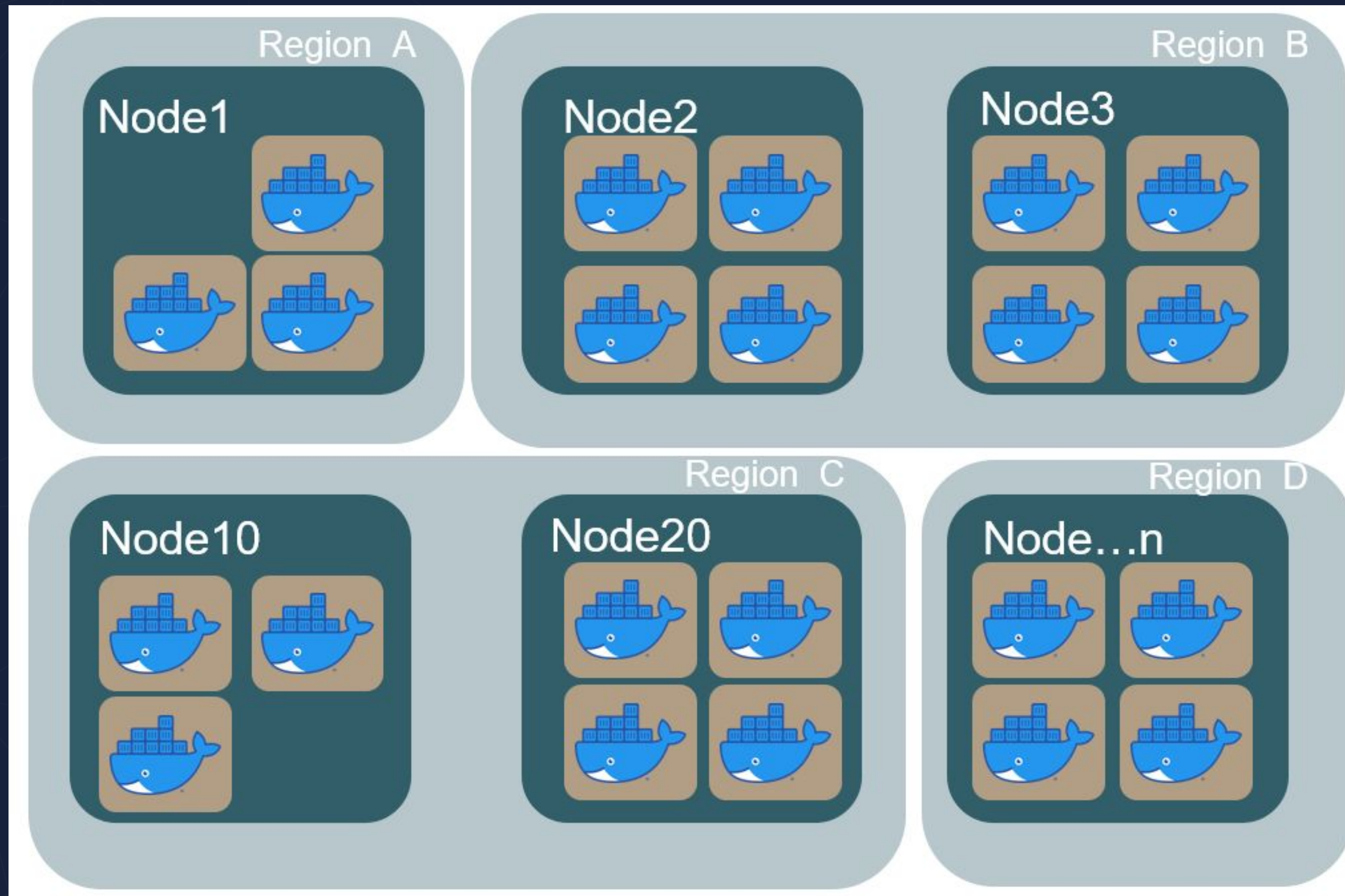
INTRODUCTION

- ▶ I am the System Architect and System Engineer with more than 15 years experience
- ▶ Cometari is a solutions company implementing DevOps culture and providing consultancy, workshops and software services.
- ▶ Our expertise are DevOps, Distributed Systems, Elastic Stack - log analysis,
- ▶ We are deeply involved in the travel tech industry
- ▶ However our solutions go much further than just integrating travel API's.

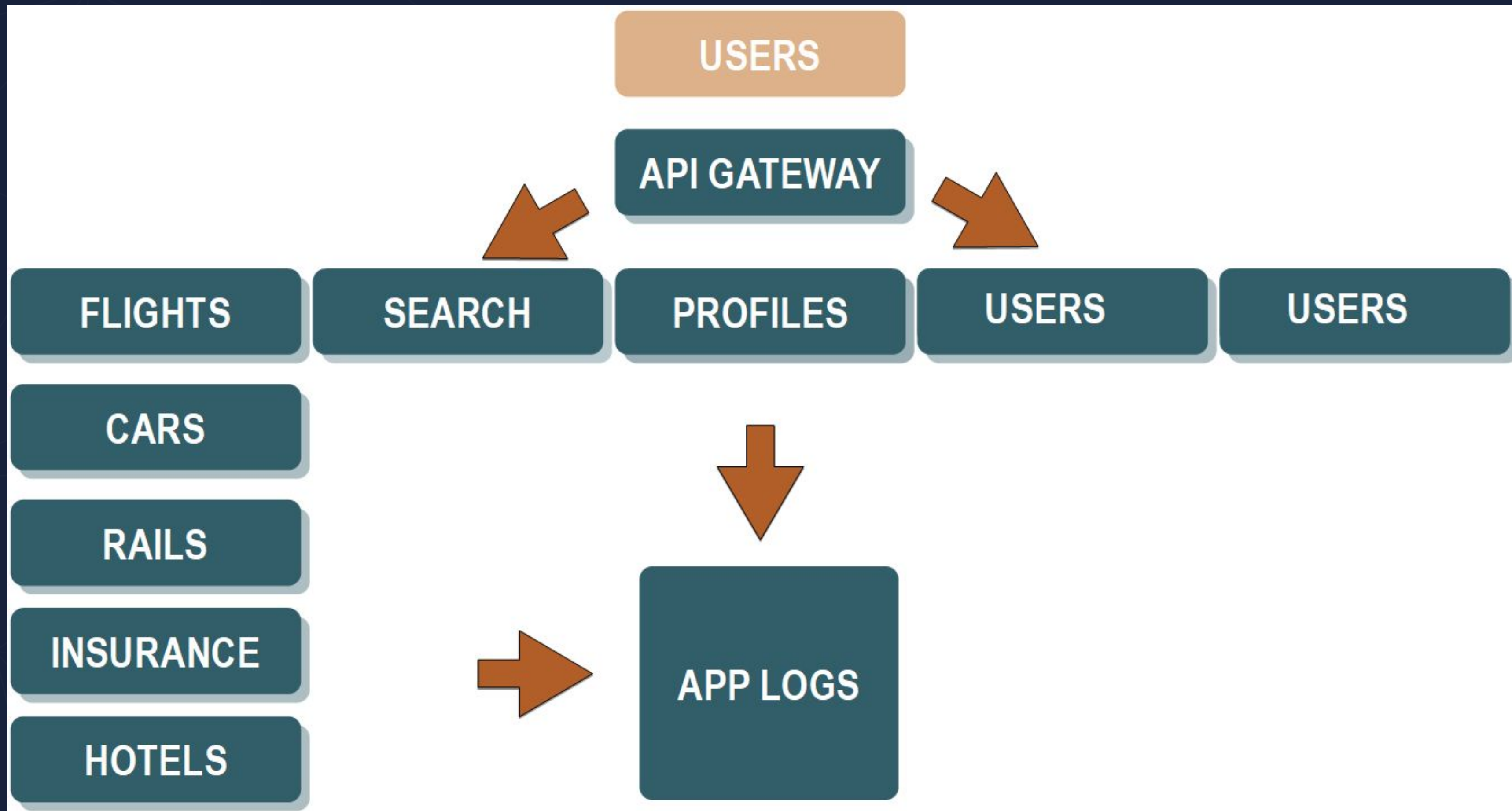


My goal is to show you how to build an effective and efficient log management system based on Elasticsearch.


CHALLENGES OF LOG ANALYSIS TODAY



CHALLENGES OF LOG ANALYSIS TODAY



CHALLENGES OF LOG ANALYSIS TODAY

kibana

Discover

Visualize

Dashboard

Timelion

Canvas

Maps

Machine Learning

Infrastructure

Logs

APM

Uptime

Dev Tools

Monitoring

Management

Logs


🔍 serviceTagName: nginx,* and nginx.status: "200"

2020-04-10 12:28:35.026	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.034	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.037	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.045	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.049	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.057	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.060	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.068	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.072	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.080	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.083	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.090	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.094	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.102	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.106	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.113	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:28:35.116	10.1.1.59 10/Apr/2020:10:28:35 +0000 POST / HTTP/1.1 200 2020041012281
2020-04-10 12:29:14.397	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012271
2020-04-10 12:29:14.401	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012271
2020-04-10 12:29:14.416	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012271
2020-04-10 12:29:14.441	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012271
2020-04-10 12:29:14.450	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012271
2020-04-10 12:29:14.456	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012271
2020-04-10 12:29:14.462	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012271
2020-04-10 12:29:14.468	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012271
2020-04-10 12:29:14.485	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012272
2020-04-10 12:29:14.501	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012273
2020-04-10 12:29:14.504	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012273
2020-04-10 12:29:14.519	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012274
2020-04-10 12:29:14.523	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012274
2020-04-10 12:29:14.540	10.1.1.51 10/Apr/2020:10:29:14 +0000 POST / HTTP/1.1 200 2020041012275

Log event document details

Field	Value
@timestamp	2020-04-10T12:27:44.723914671+02:00
_id	NTg3YjEyYjktNjY3OC00N2U5LTImZTetMzcyMWI0NGU3MDdj
_index	docker-logs-2020.04
container_id	d5fb8c03aa3a1f06862ca28933268c5a1aeb9530f960f035e8275c1089e2ed29
container_name	/etl_nginx.1.yz8yq01c0h2i9l0gu0wu3s3p4
loglevel	INFO
message	10.1.1.53 10/Apr/2020:10:27:44 +0000 POST / HTTP/1.1 200 2020041012273953000 Java/1.8.0_241
nginx.body_bytes_sent	2
nginx.http_LogId	2020041012273953000
nginx.http_referrer	
nginx.http_user_agent	Java/1.8.0_241
nginx.remote_addr	10.1.1.53
nginx.remote_user	
nginx.request	POST / HTTP/1.1
nginx.request_time	0.002
nginx.status	200

CHALLENGES OF LOG ANALYSIS TODAY

 **kibana**

Discover

Visualize

Dashboard

Timelion

Canvas

Maps

Machine Learning

Infrastructure

Logs

APM

Uptime

Dev Tools

Monitoring

Management

Logs

🔍

message:"Delete indices ('docker-logs-*'"

No additional entries found

2020-01-01 01:30:08.588	Trying Action ID: 1, "delete_indices": Delete indices ('docker-logs-*
2020-02-01 01:30:10.712	Trying Action ID: 1, "delete_indices": Delete indices ('docker-logs-*
2020-03-01 01:30:10.264	Trying Action ID: 1, "delete_indices": Delete indices ('docker-logs-*
2020-04-01 01:30:10.947	Trying Action ID: 1, "delete_indices": Delete indices ('docker-logs-*

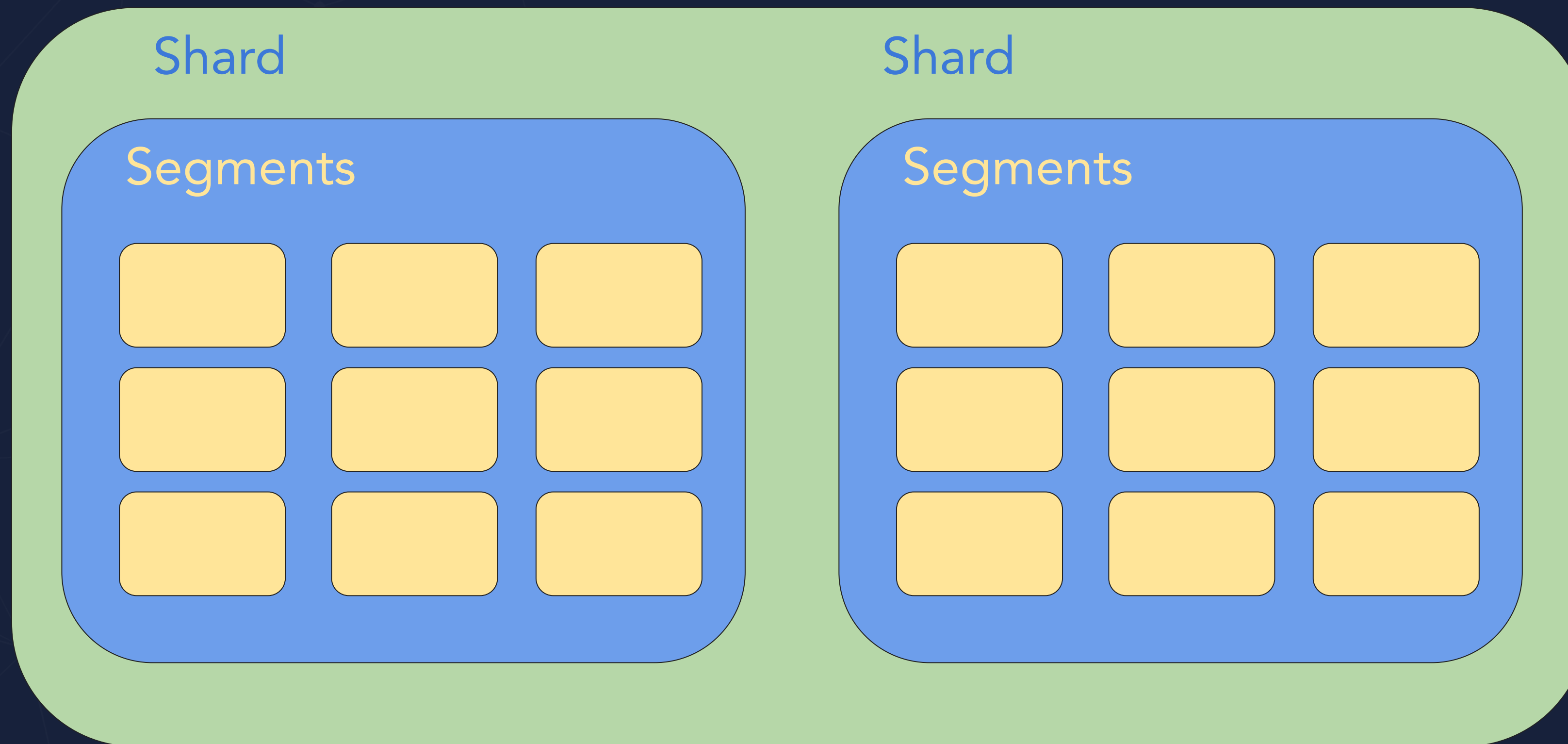
No additional entries found [↻ Load again](#)

Log event document details

Field	Value
@timestamp	🔍 2020-04-01T01:30:10.947835064+02:00
_id	🔍 MTJmNmU2MjMtZmM0ZS00M2FILWFIMDAzMzJlZTJlZDA0ODc
_index	🔍 docker-logs-2020.03
container_id	🔍 a3cbafa08bc6670f249fa9f57756d0f6ab24e1e25eb1df81902138341a30d
container_name	🔍 /house_keeping_curator.1.9nqodlbd504paj90slzw8hahc
loglevel	🔍 INFO
message	🔍 Trying Action ID: 1, "delete_indices": Delete indices ('docker-lo *') older than 4 months.
serviceTagName	🔍 curator
source	🔍 stdout

INDICES, SHARDS, SEGMENTS

Index



Logs - good practises

- ▶ Normalize your logs (e.g. use the JSON structure)
- ▶ Use the same time zone in logs (within the entire services)
- ▶ Take care about the sensitive data
- ▶ Use the correlation identifiers (very useful in distributed systems)
- ▶ Log only important/useful data

PLANNING CLUSTER AND DESIGNING INDICES

- ▶ What your logs look like
- ▶ The logs size: more or less constant or various
- ▶ Optimal size of shard
- ▶ How many primary shards/replicas?
- ▶ Indices: daily, weekly, monthly, fixed size?
- ▶ How you are going to search your data?
- ▶ Index templates
- ▶ Aliases

Index template

- ▶ refresh_interval
- ▶ disable dynamic mapping
- ▶ use the right data type

Schema for common logs

Field name	data type	mandatory
@timestamp	date	true
message	text	true
serviceTagName	text/keyword	true
loglevel	keyword	true
traceld	keyword	false
spanId	keyword	false
<specific_service>	JSON object	false

Index template

```
{
  "index_patterns": ["docker-logs-*"],
  "version": 1,
  "settings": {
    "number_of_shards": 1,
    "number_of_replicas": 0,
    "refresh_interval": "10s"
  },
  "mappings": {
    "_doc": {
      "dynamic": "false",
      "properties": {
        "@timestamp": {
          "type": "date"
        },
        "message": {
          "type": "text"
        },
        "serviceTagName": {
          "type": "text",
          "fields": {
            "keyword": {
              "type": "keyword"
            }
          }
        },
        "loglevel": {
          "type": "keyword"
        },
        "nginx": {
          "properties": {
            "http_LogId": {
              "type": "text",
              "fields": {
                "keyword": {
                  "type": "keyword"
                }
              }
            },
            "status": {
              "type": "keyword"
            }
          }
        }
      }
    }
  },
  "aliases": {
    "docker-logs": {}
  }
}
```

How to start?

- ▶ create a initial version of your index (based on planning)
- ▶ start indexing your data
- ▶ start monitoring size of your shards, resources, cluster
- ▶ check the queries/aggregations speed
- ▶ tune your cluster/indices/settings
- ▶ test and validate your solution!

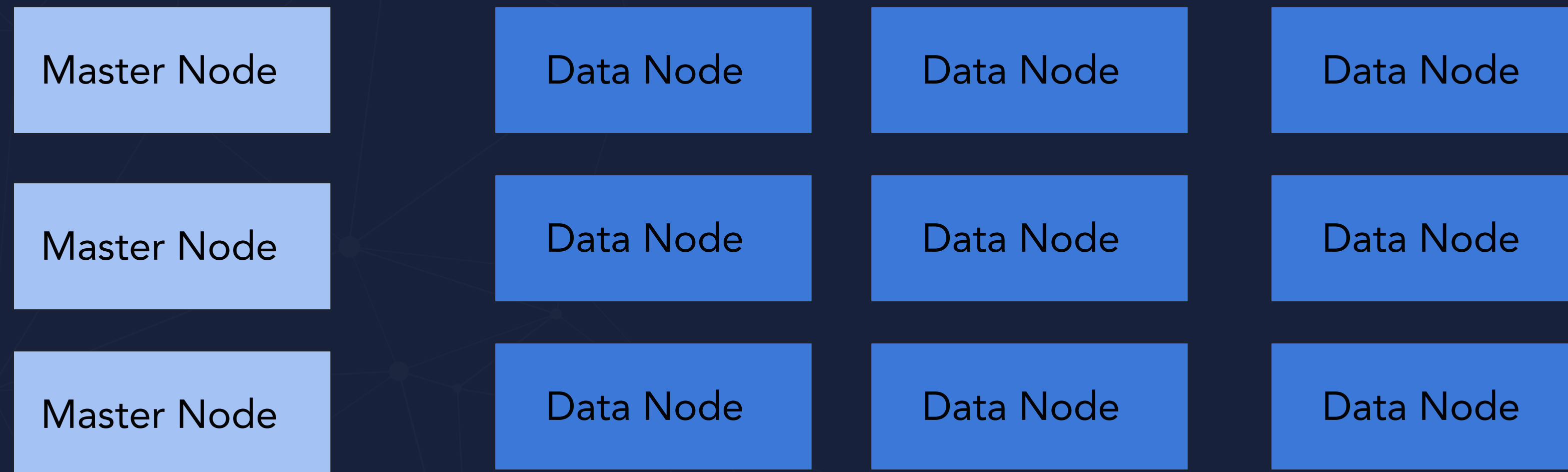


Start using your first configuration on production, monitor your system and tune the settings if necessary. Your cluster should live and should be constantly improved.

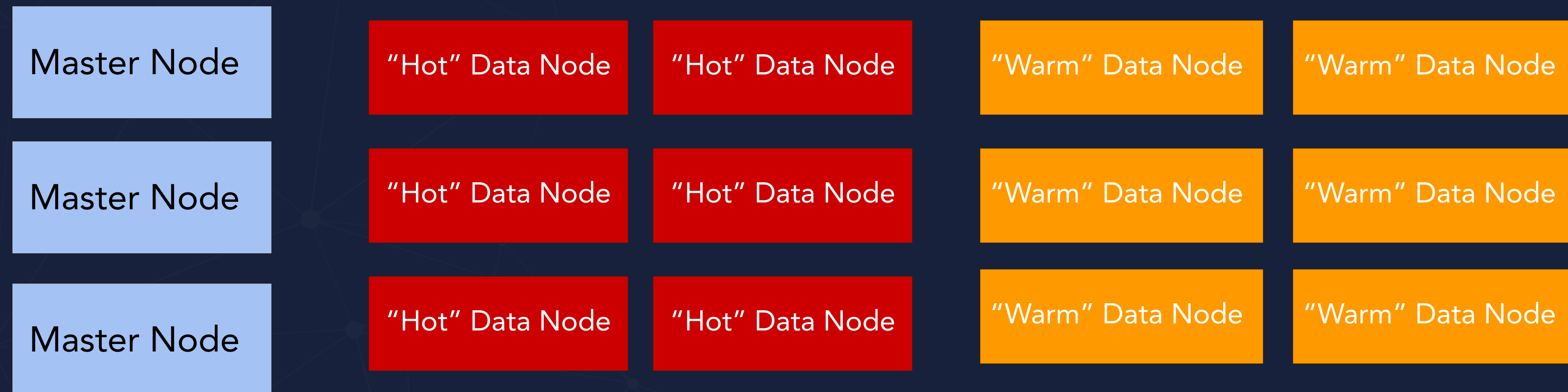


ARCHITECTURES OF DEPLOYMENTS

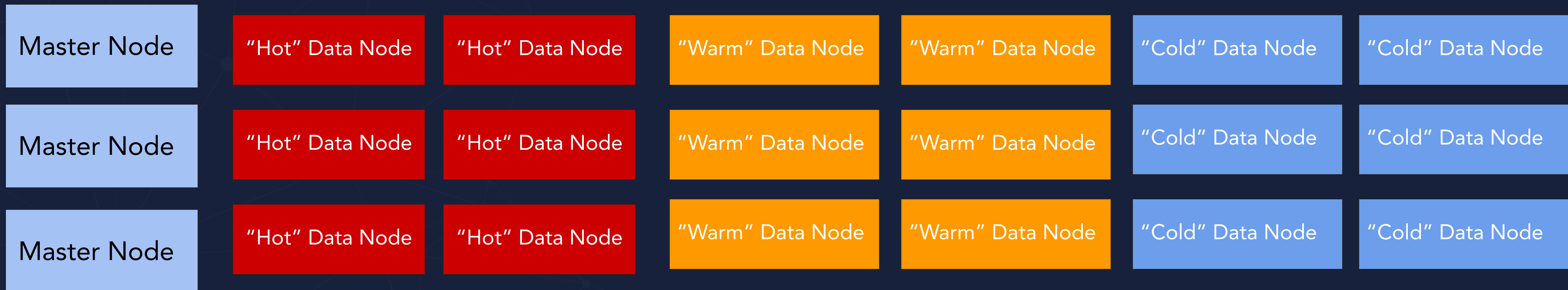
Uniform architecture



Hot-warm architecture



Hot-warm-cold architecture





BUILDING AN EFFICIENT LOG SYSTEM AT SCALE - FOR FREE!

Split your indices to “hot” and “warm”

- ▶ “hot” indices - the latest ones: writable and searchable
- ▶ “warm” - read-only, contains older logs
- ▶ the right amount of the primary shards (more for “hot”, less for “warm”)
- ▶ splitting based on “writing time range”

Shrink the shards

- ▶ move one copy of shards to a single node

```
PUT index_name/_settings
{
  "index.blocks.write": true,
  "index.routing.allocation.require._name": "some_node_name"
}
```

- ▶ do the shrink

```
POST index_name/_shrink/index_name-warm
{
  "settings": {
    "number_of_shards": 1,
    "number_of_replicas": 0,
    "codec": "best_compression",
    "index.blocks.write": true,
    "index.routing.allocation.require._name": null
  }
}
```

Force a merge of segments

```
POST index_name-warm/_forcemerge?max_num_segments=1
```


Increase the number of replicas

```
PUT index_name-warm/_settings
{
  "number_of_replicas": 1
}
```

Switch aliases

```
POST _aliases
{
  "actions": [
    {
      "remove": {
        "index": "index_name",
        "alias": "logs-view"
      }
    },
    {
      "add": {
        "index": "index_name-warm",
        "alias": "logs-view"
      }
    }
  ]
}
```

Remove the "hot" indices

```
DELETE index_name
```



CURATOR - MANAGING TOOL

CURATOR -



A screenshot of a file explorer window displaying a list of files related to the Curator tool. The files are listed in a vertical column, each preceded by a small icon representing a document. The first file, 'crontab_jobs', is highlighted with a light gray background. The other files are 'curator.yml', 'delete_docker_logs_indices_action.yml', 'delete_indices_action.yml', 'delete_missing_logs_indices_action.yml', 'rollover_missing_logs_indices_action.yml', and 'shrink_force_marge_indices_action.yml'.

- crontab_jobs
- curator.yml
- delete_docker_logs_indices_action.yml
- delete_indices_action.yml
- delete_missing_logs_indices_action.yml
- rollover_missing_logs_indices_action.yml
- shrink_force_marge_indices_action.yml

CURATOR -

```
actions:
  1:
    action: delete_indices
    description: >-
      Delete indices ('docker-logs-*) older than 4 months.
    options:
      ignore_empty_list: True
      continue_if_exception: True
      disable_action: False
    filters:
      - filtertype: pattern
        kind: prefix
        value: docker-logs-
        exclude:
      - filtertype: age
        source: name
        direction: older
        timestring: '%Y.%m'
        unit: months
        unit_count: 4
```

SUMMARY

- ▶ Take time for good planning
- ▶ Normalize your logs
- ▶ Use the same time zone for all logs (UTC is more than welcome)
- ▶ You must know your logs
- ▶ The right balance between the number and size of shards
- ▶ Choose the right solution architecture
- ▶ Monitor your cluster (shard size, heap space size, etc.)
- ▶ Automate the managing your logs (e.g. curator)
- ▶ Don't forget to add alerting!

RAFAL.BAJEK@COMETARI.COM

THANK YOU