



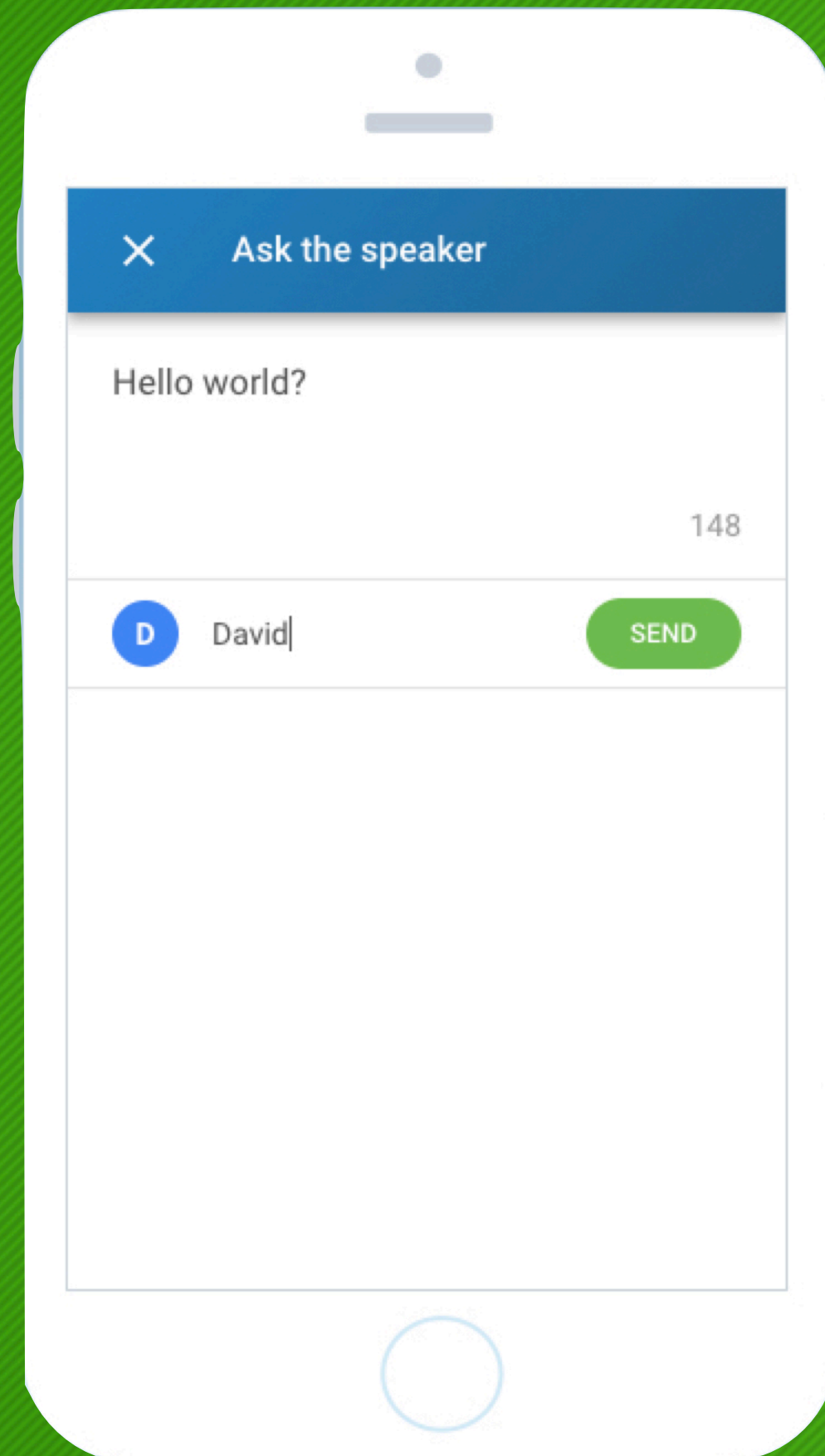
Les vendredi noirs ? Même pas peur !

David Pilato
Developer | Evangelist, @dadoonet

VOXXEDDAYS
LUXEMBOURG



slido



Join at
slido.com
#elastic



Data Platform Architectures

fridge:leftovers

user:soulmate

life:universe

_Search? outside the box

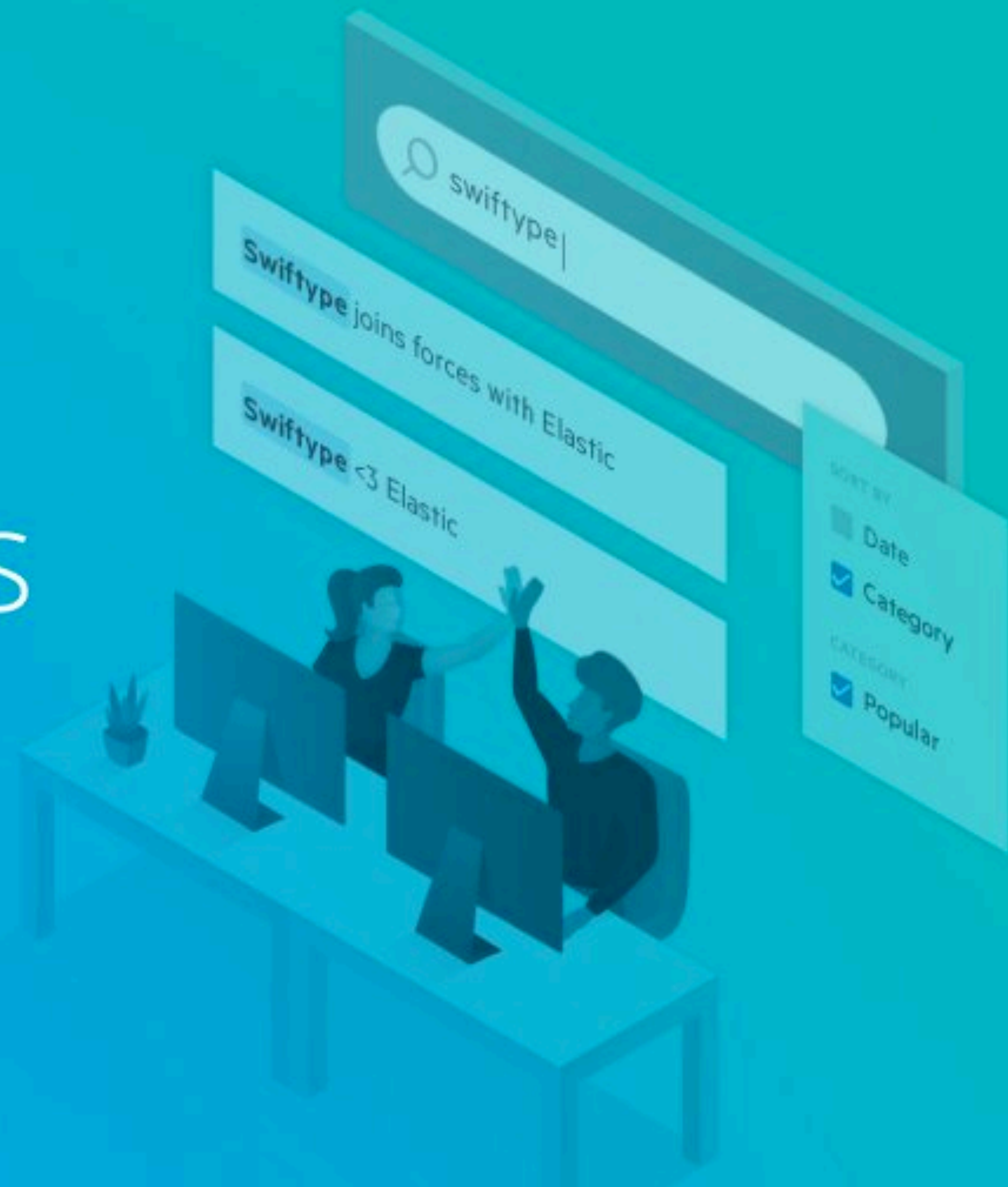
city:restaurant

car:model

work:dreamjob



Elastic Joins Forces with Swifttype

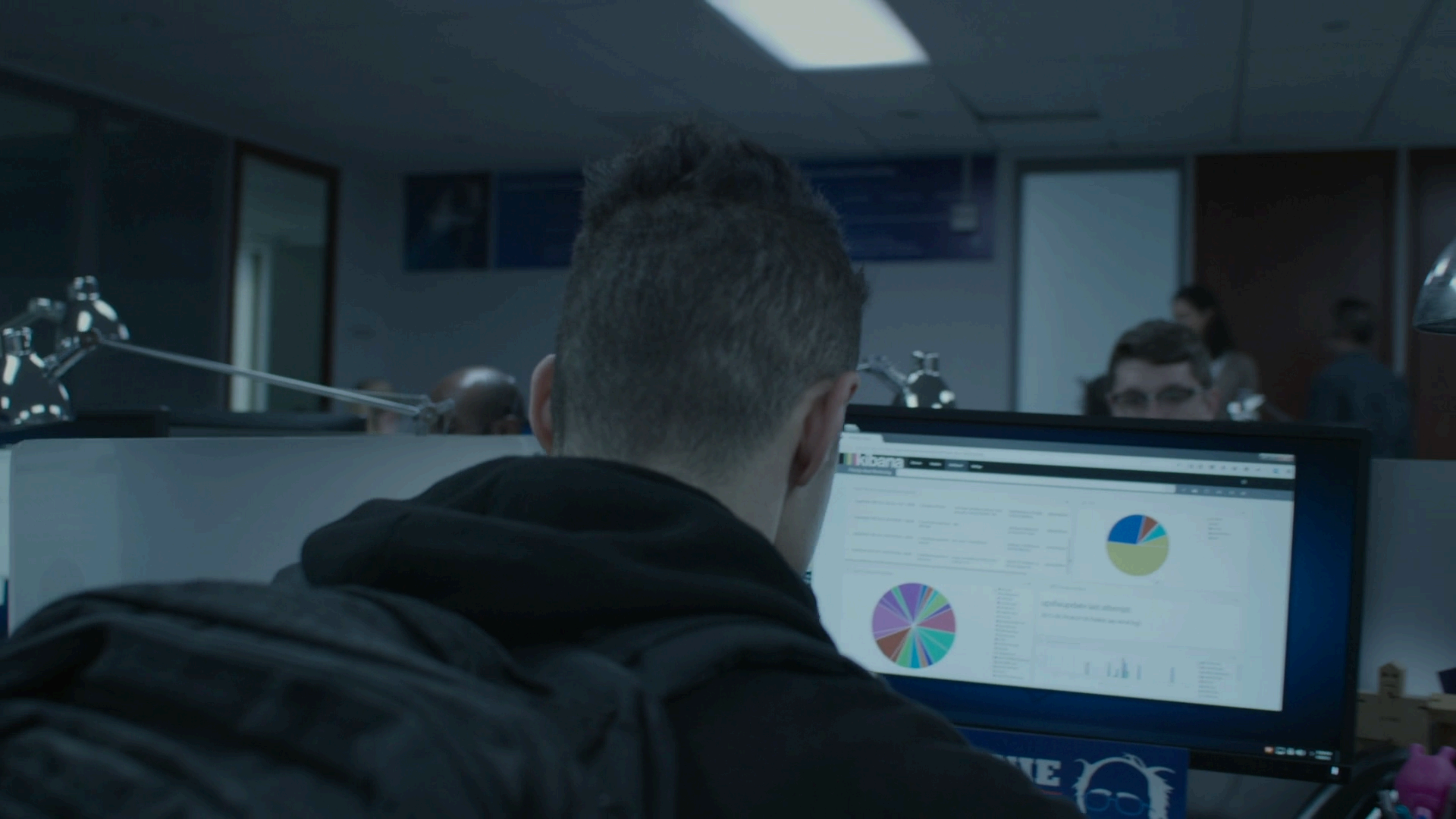


Logging

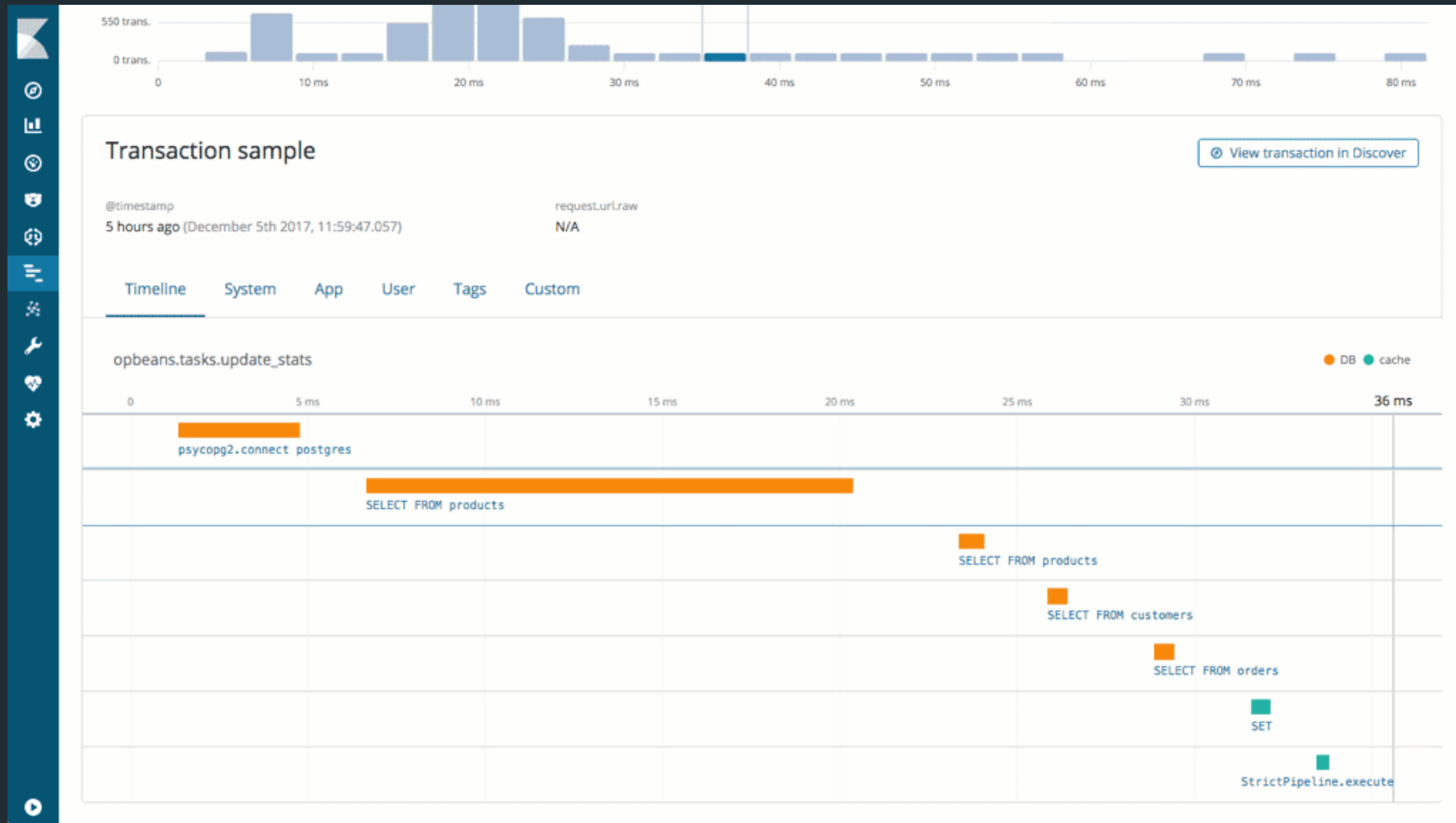


Metrics

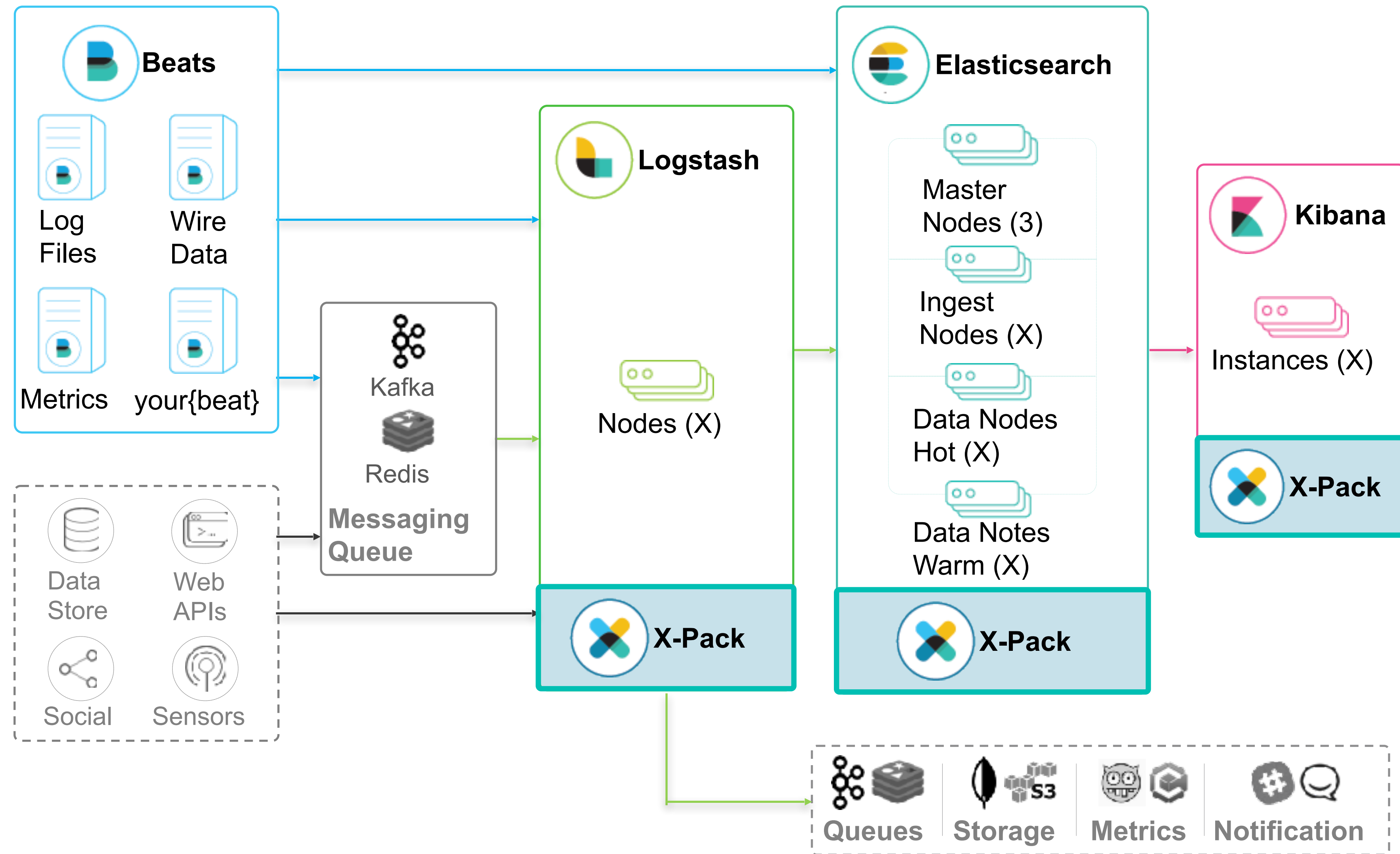




APM



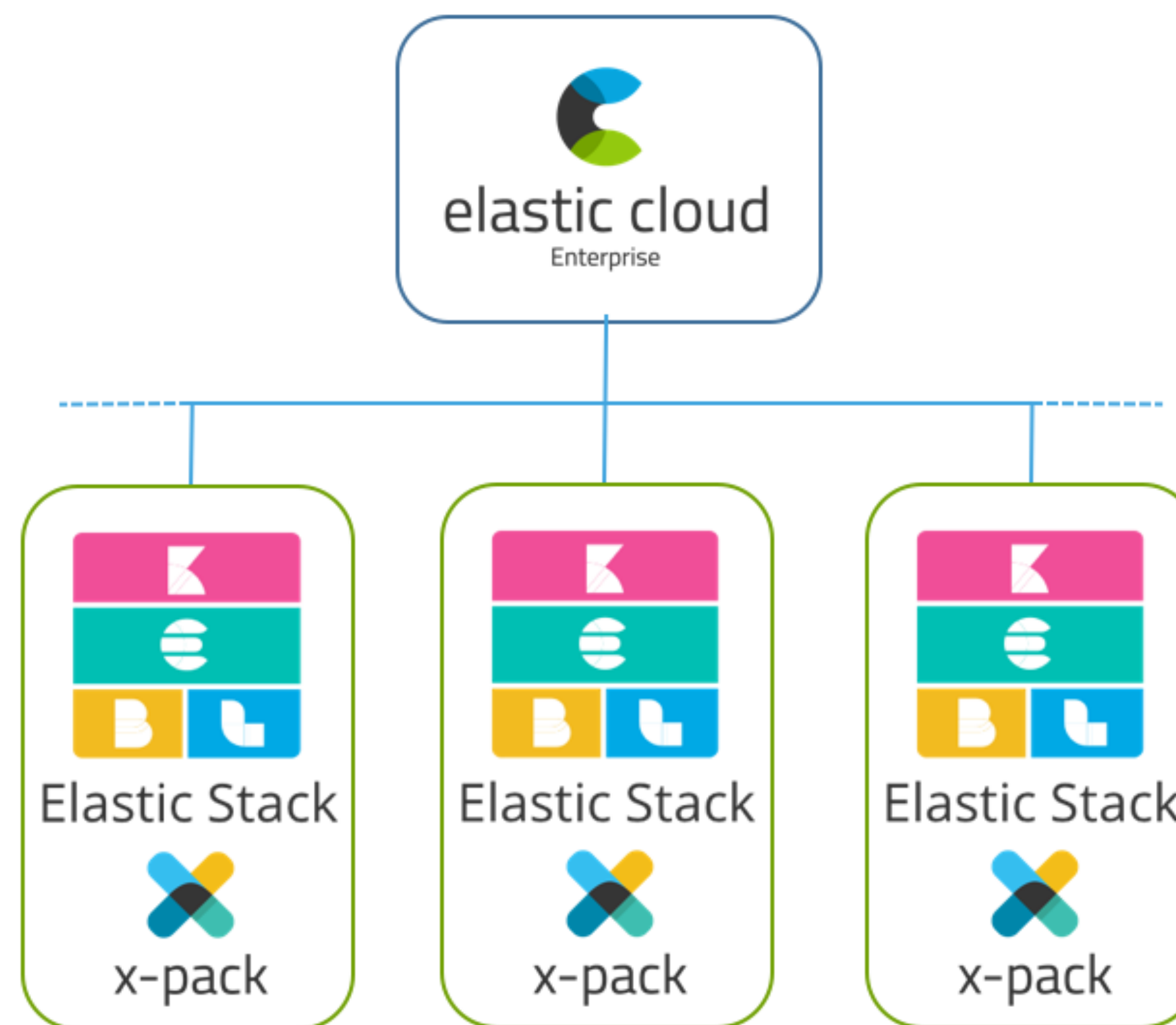
The Elastic Journey of Data





cloud Enterprise

Provision and manage multiple Elastic Stack environments and provide **search-aaS, logging-aaS, BI-aaS, data-aaS** to your entire organization





Hosted Elasticsearch & Kibana
Includes X-Pack features
Starts at \$45/mo

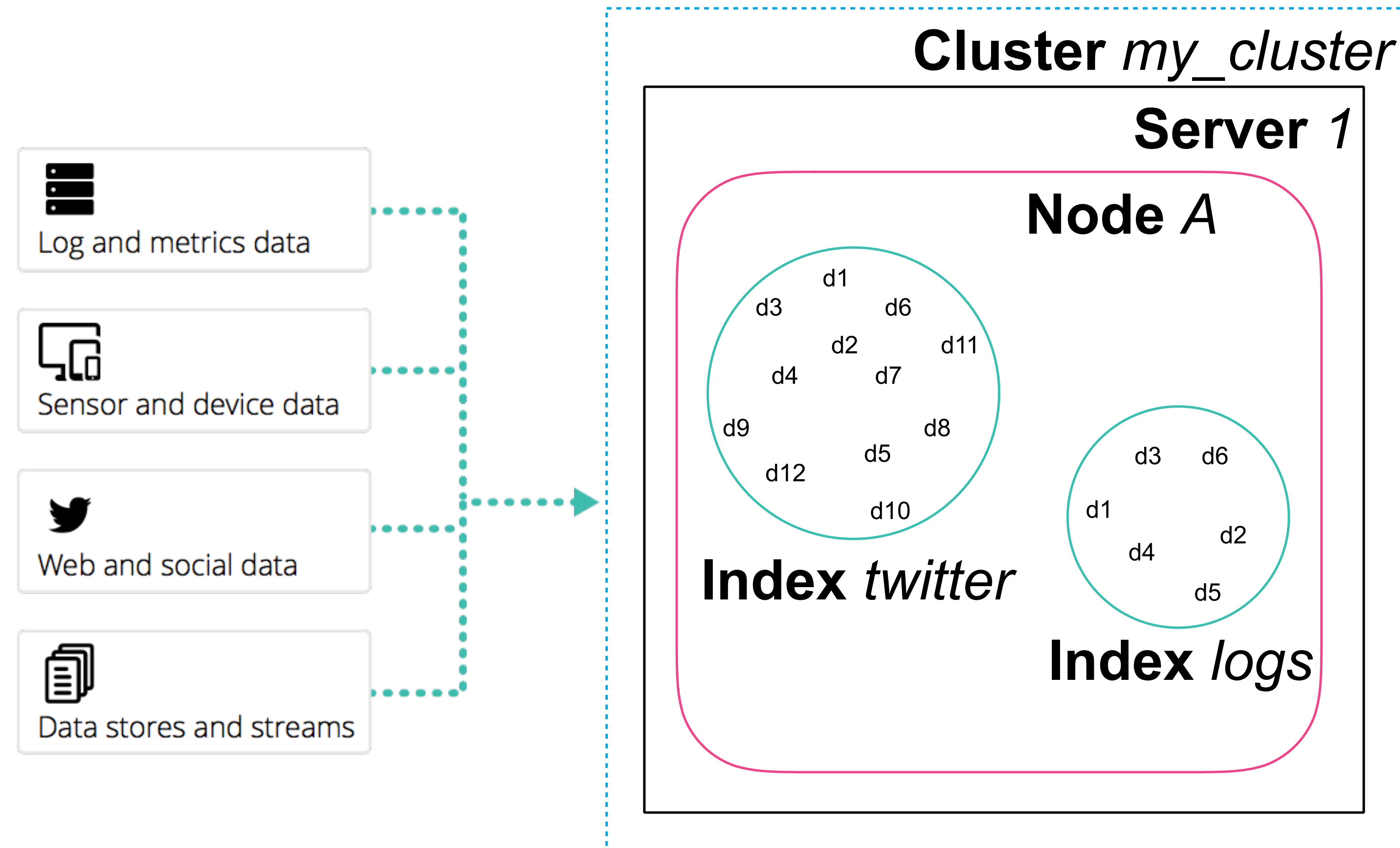
A screenshot of the Elastic Cloud console interface. The top navigation bar includes 'cloud', 'Clusters', 'Plugins', 'Help', 'Account', and 'Sign out'. The main content area is divided into two columns. The left column, titled 'Summary', contains a table with configuration details: Platform (Amazon Web Services), Region (US East (N. Virginia)), Memory (1 GB), Storage (24 GB), SSD (Yes), High availability (No), Hourly rate (\$0.0612), and Monthly rate (\$45). Below the table is a blue 'Create' button. The right column, titled 'Cluster Size', features a slider to choose a cluster size, with a legend for Memory (blue) and Storage (light blue). The slider is set to 4GB Memory and 96GB Storage. Below the slider, there is a checkbox for 'SSD' which is checked, and a link 'Need a larger cluster? Contact us.' The bottom section, titled 'Cloud Platform', shows 'Pick your cloud:' with buttons for 'amazon web services' and 'Google Cloud Platform'. Below this, 'Choose a region near you:' shows buttons for 'US East (N. Virginia)', 'US West (N. California)', 'US West (Oregon)', 'EU (Ireland)', 'Asia Pacific (Singapore)', 'Asia Pacific (Tokyo)', 'South America East', and 'Asia Pacific (Sydney)'.

Available in
Amazon Web Service
Google Cloud Platform

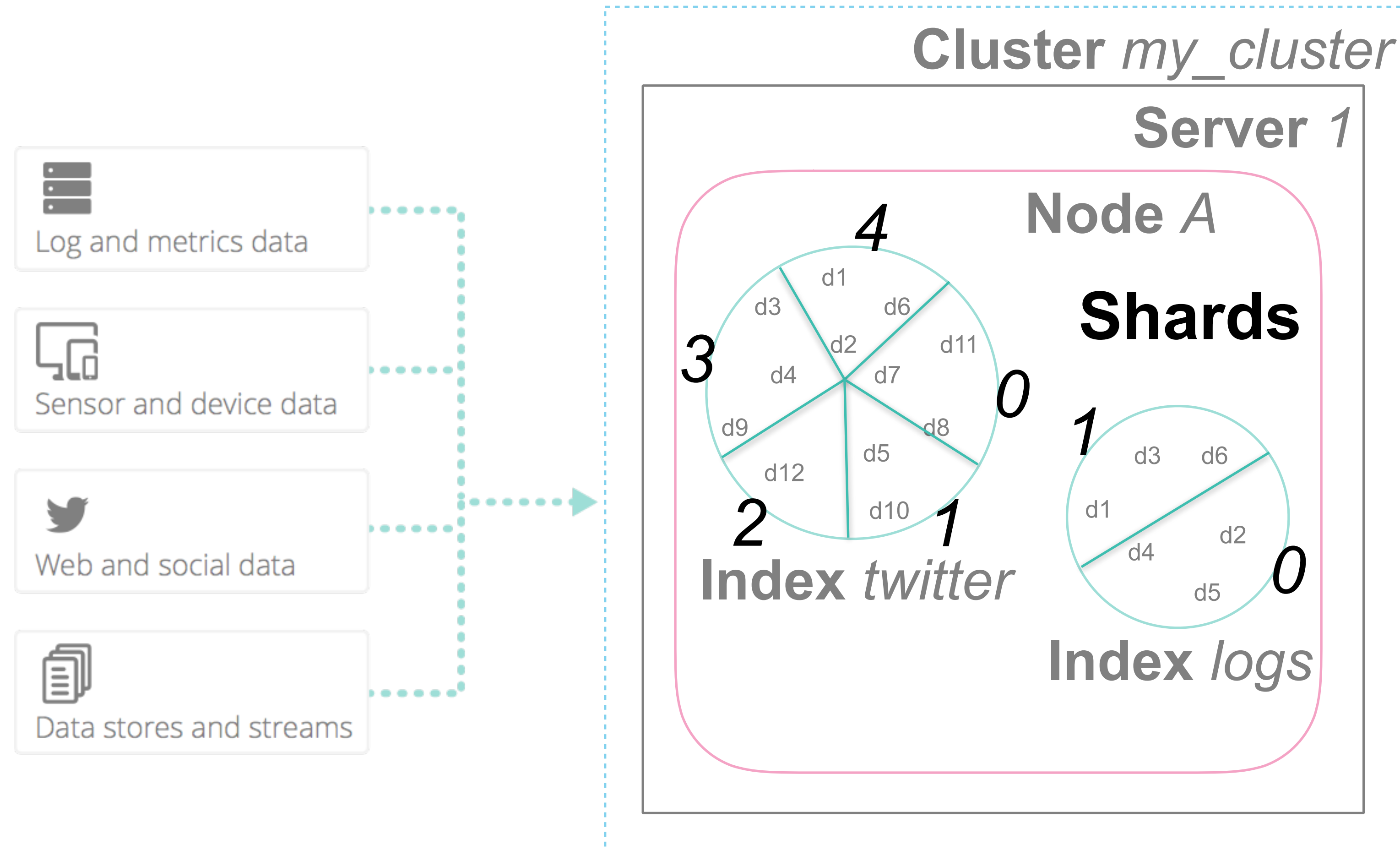


Elasticsearch Cluster Sizing

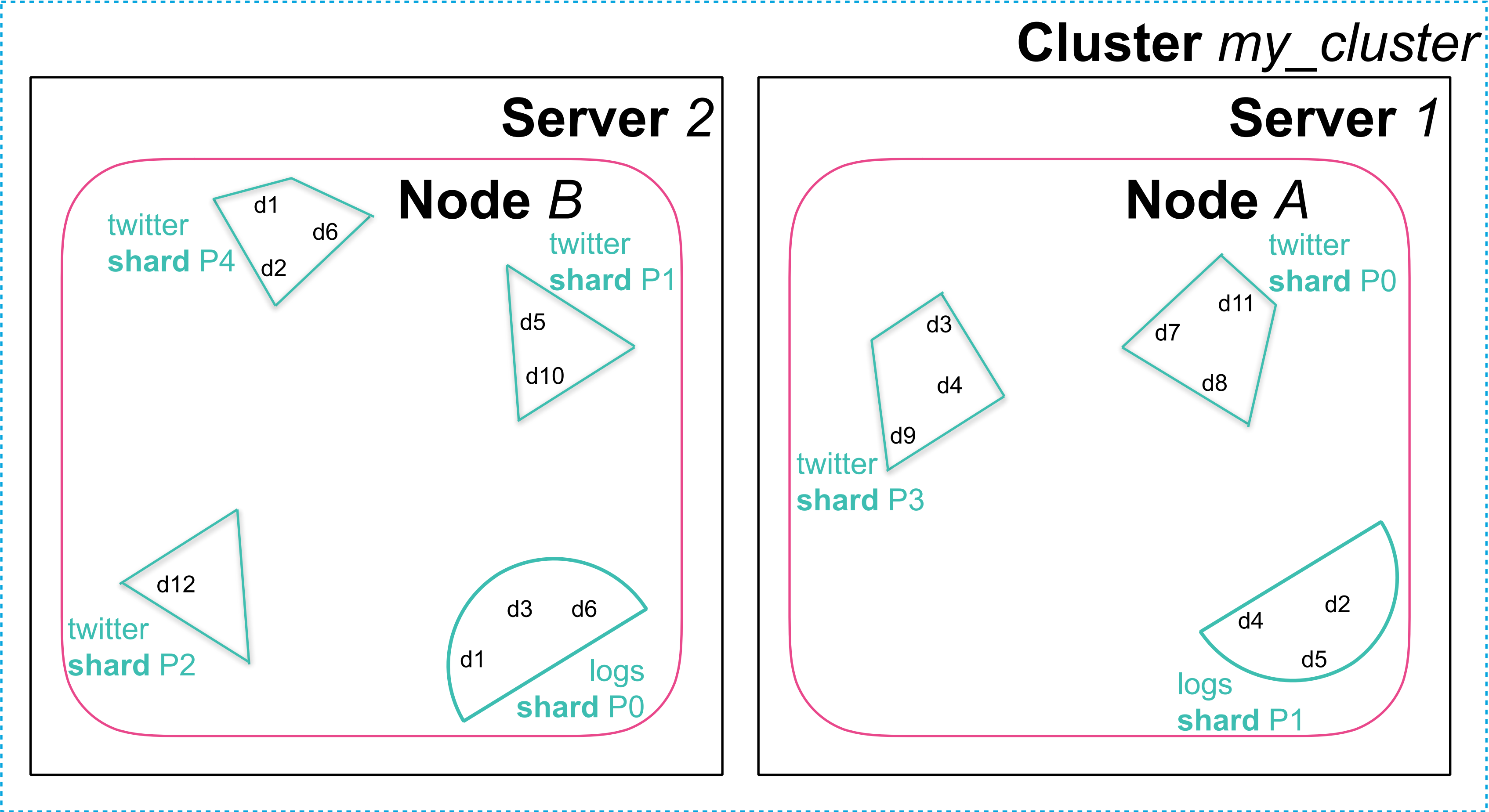
Terminology



Partition

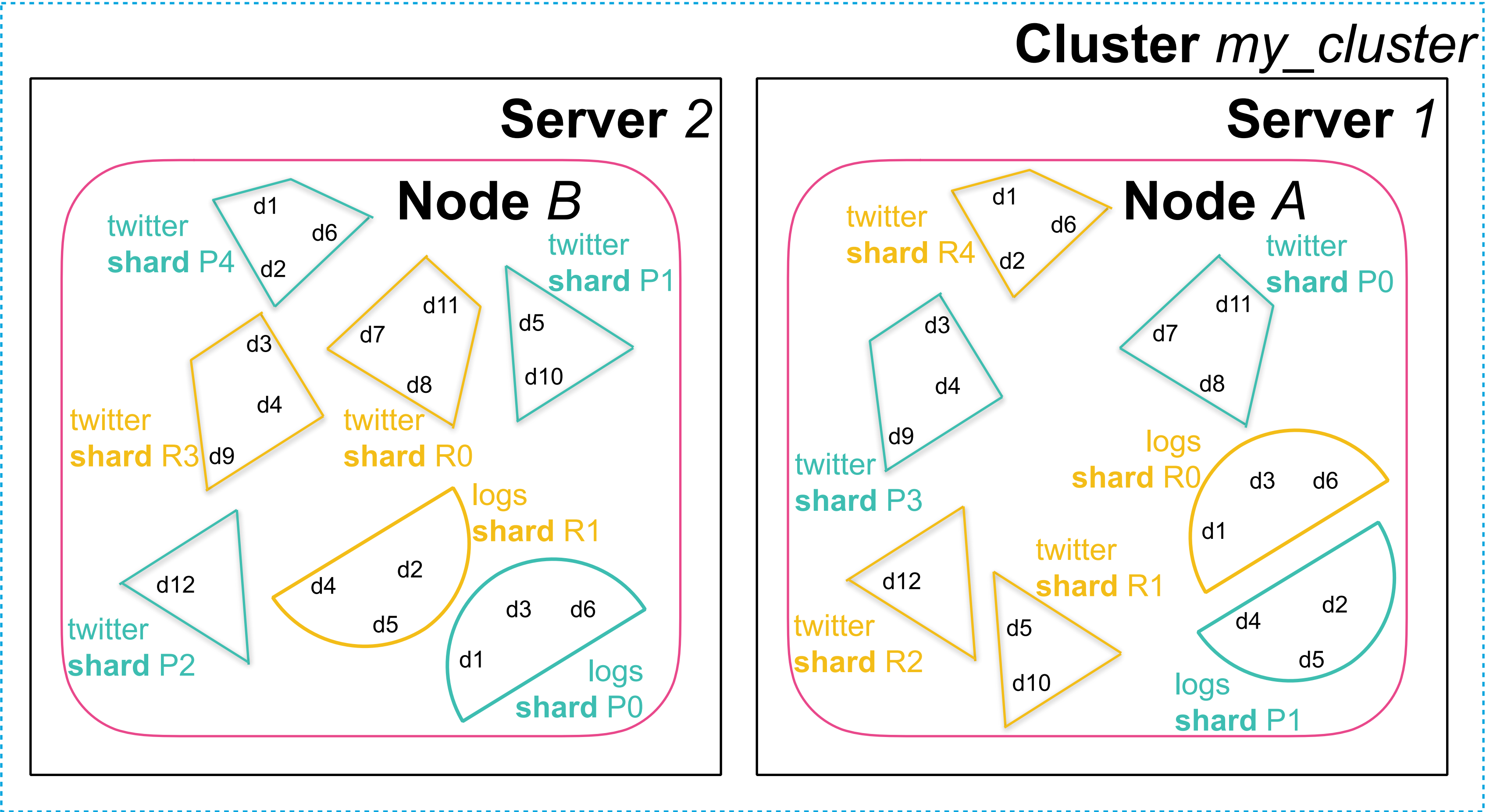


Distribution



Replication

- Primaries
- Replicas



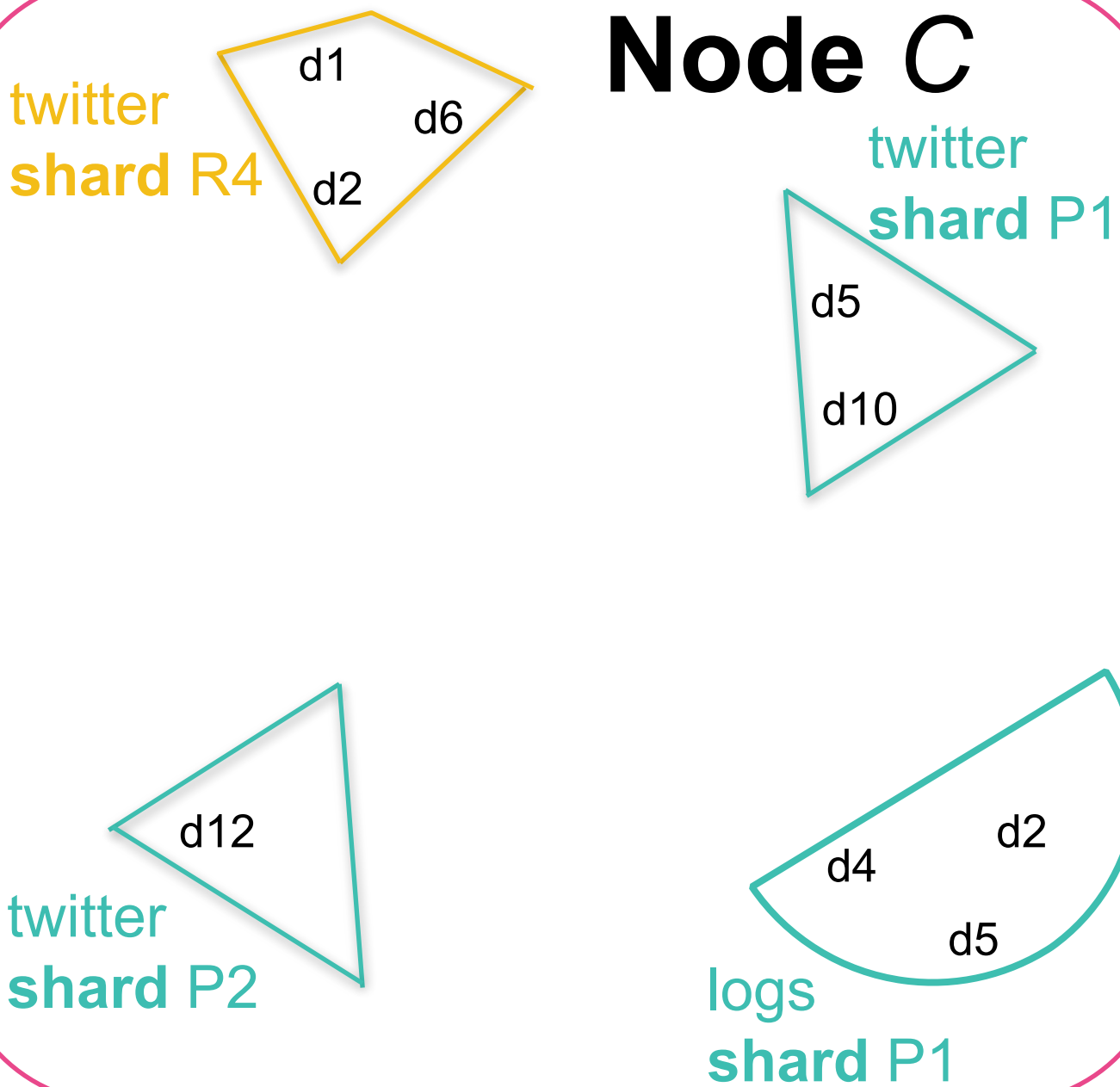
Replication

- Primaries
- Replicas

Cluster *my_cluster*

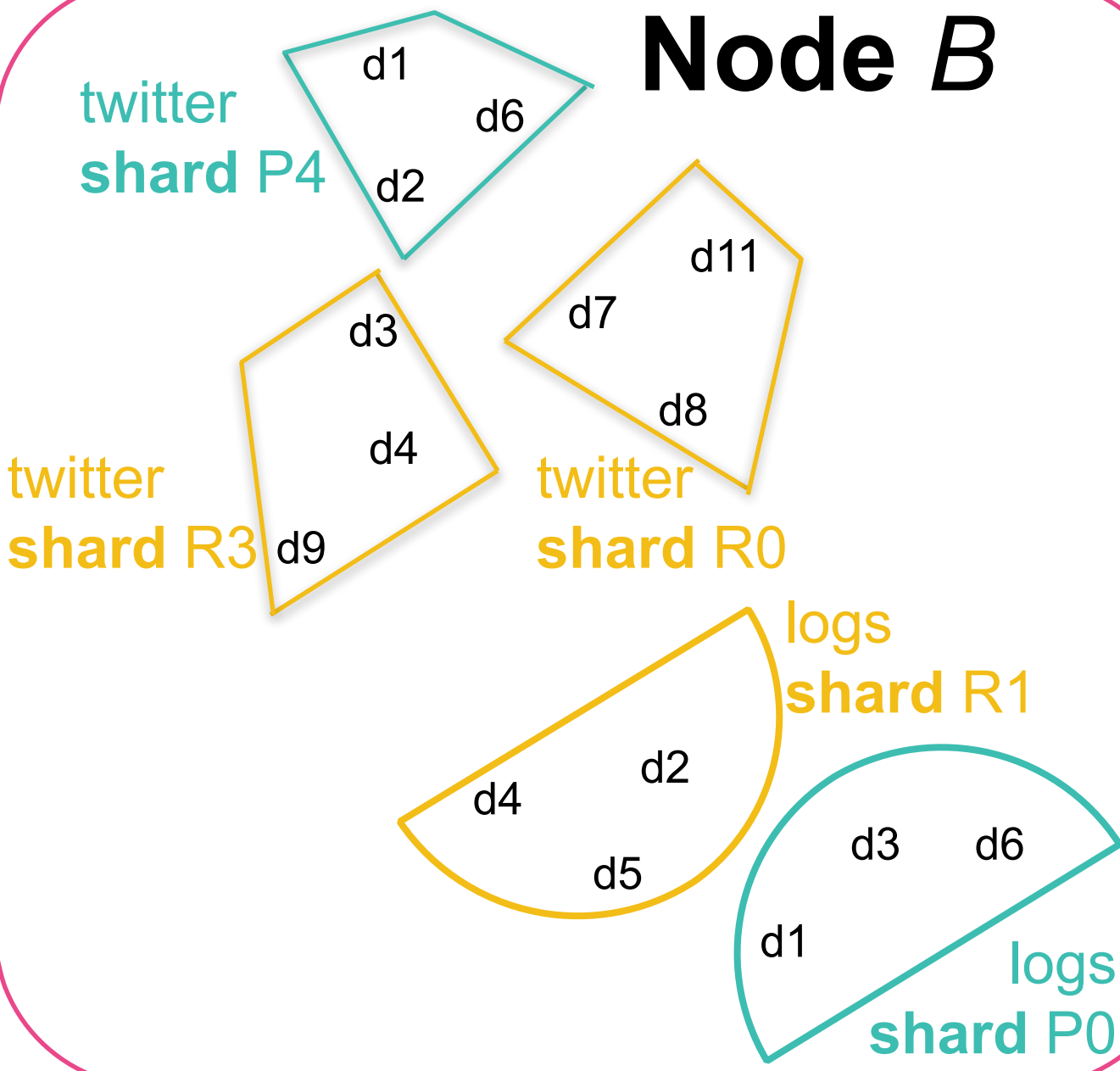
Server 3

Node C



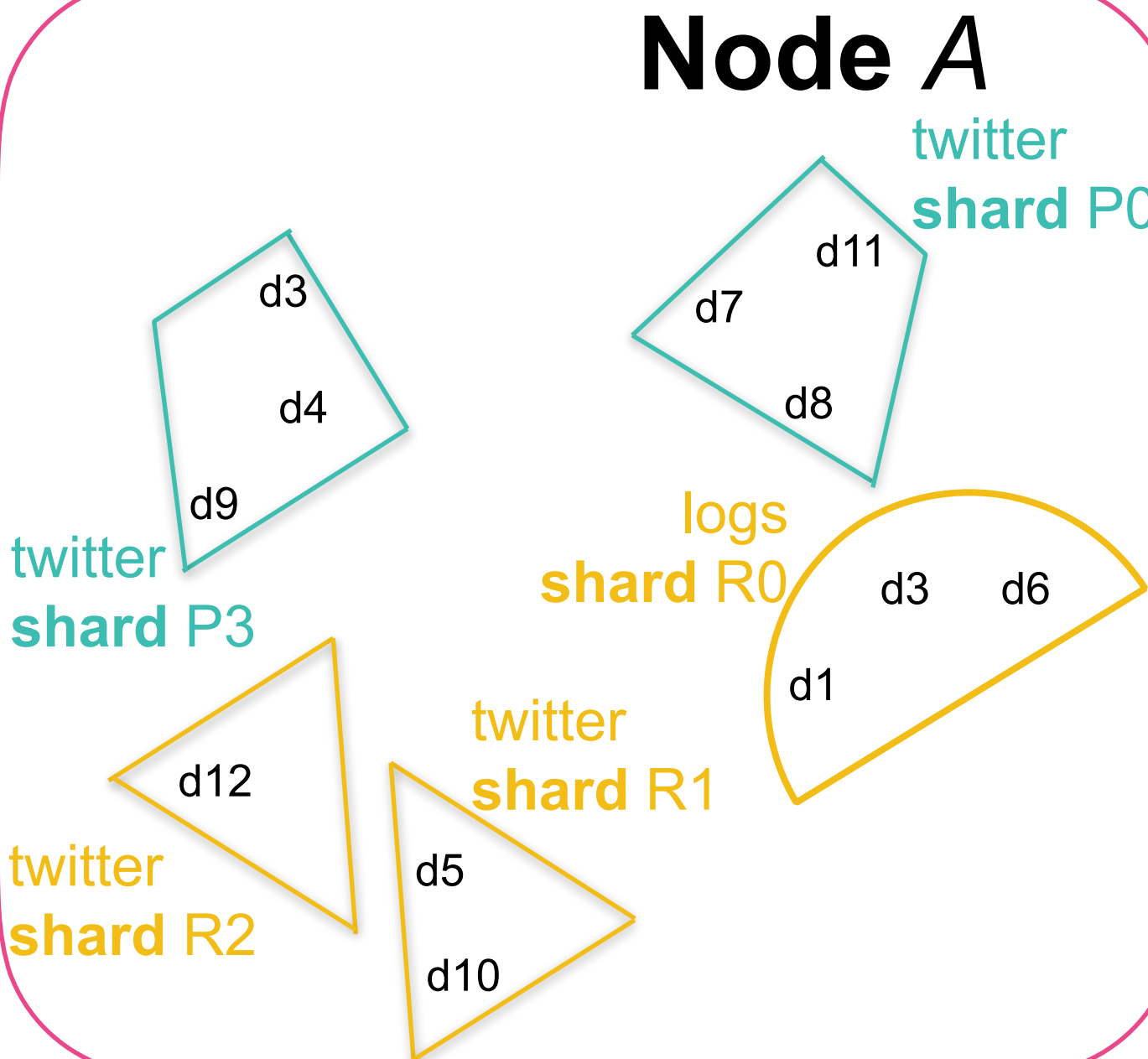
Server 2

Node B



Server 1

Node A



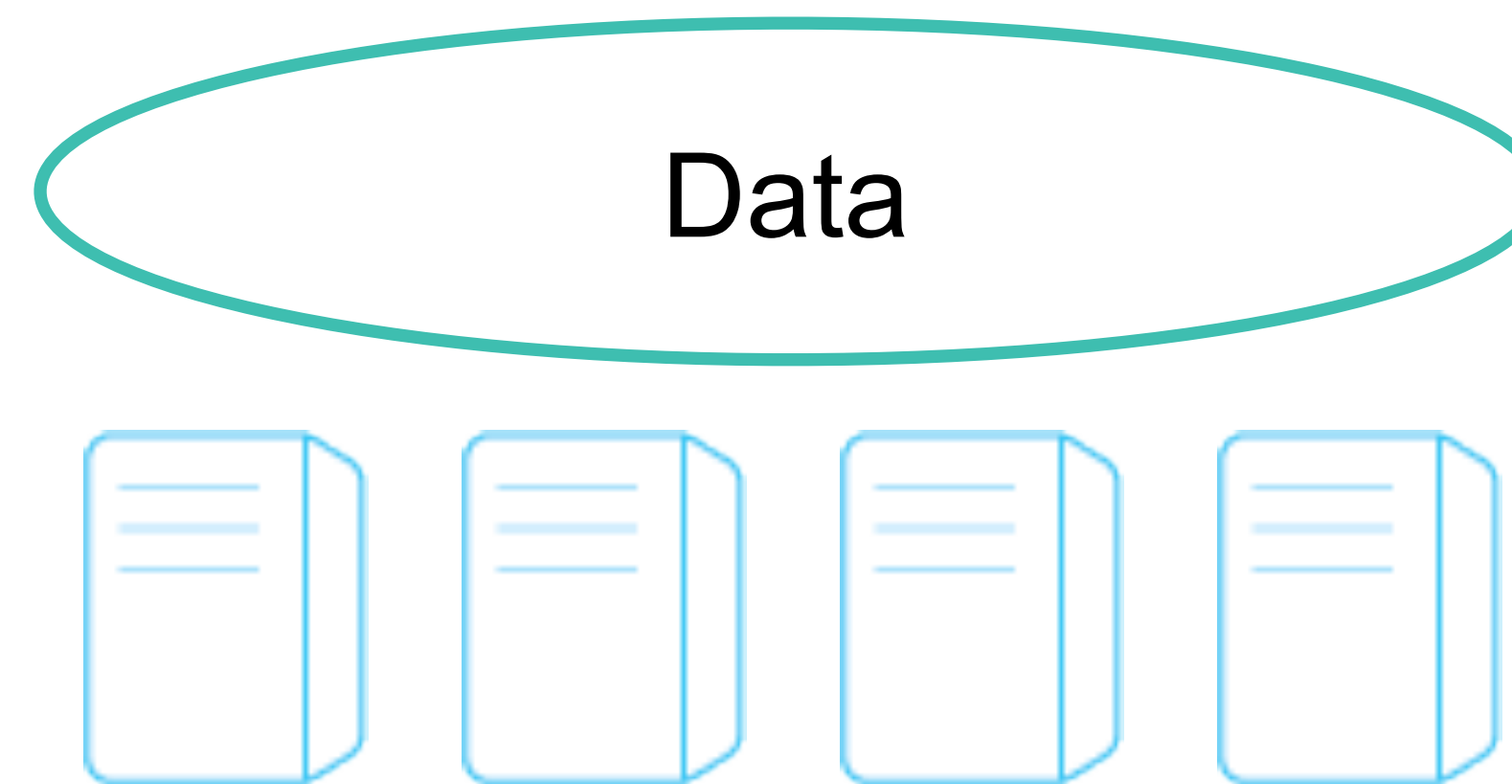
Scaling



Scaling



Scaling



Scaling



Scaling



- In Elasticsearch, **shards** are the **working unit**
- More data -> More shards

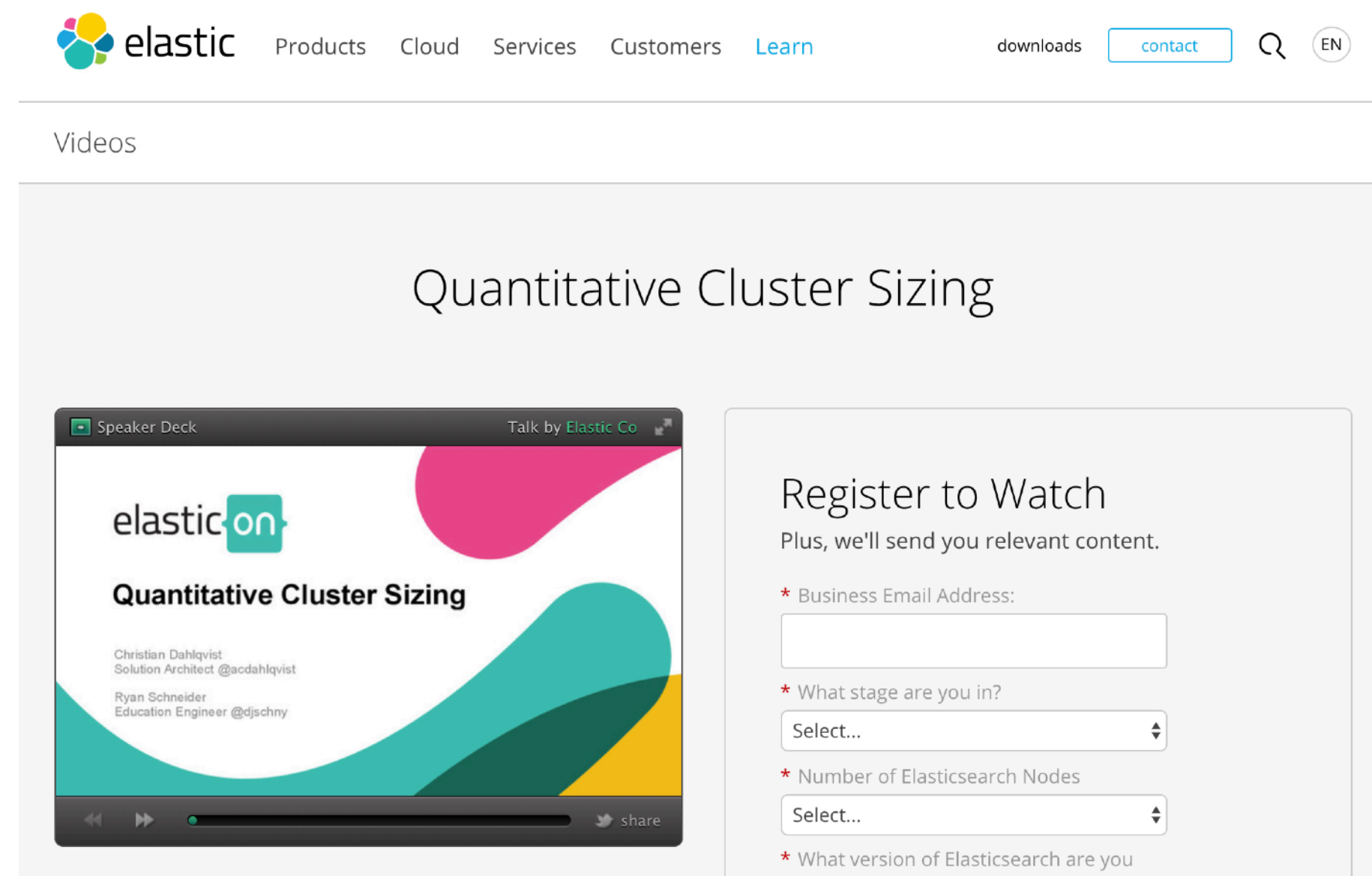
But how many shards?

How much data?

- ~1000 events per second
- $60s * 60m * 24h * 1000 \text{ events} \Rightarrow \sim 87M \text{ events per day}$
- 1kb per event $\Rightarrow \sim 82GB \text{ per day}$
- 3 months $\Rightarrow \sim 7TB$

Shard Size

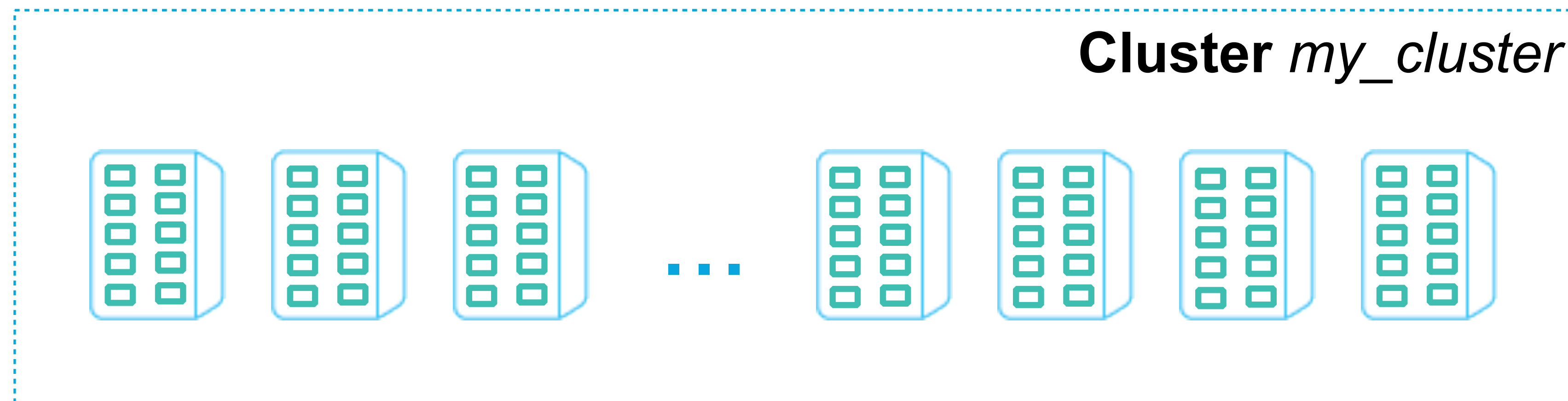
- It depends on many different factors
 - document size, mapping, use case, kinds of queries being executed, desired response time, peak indexing rate, budget, ...
- After the shard sizing*, each shard should handle **45GB**
- Up to 10 shards per machine



* <https://www.elastic.co/elasticon/conf/2016/sf/quantitative-cluster-sizing>

How many shards?

- Data size: ~7TB
- Shard Size: ~45GB*
- **Total Shards: ~160**
- Shards per machine: 10*
- **Total Servers: 16**



* <https://www.elastic.co/elasticon/conf/2016/sf/quantitative-cluster-sizing>

But...

- How many indices?
- What do you do if the daily data grows?
- What do you do if you want to delete old data?

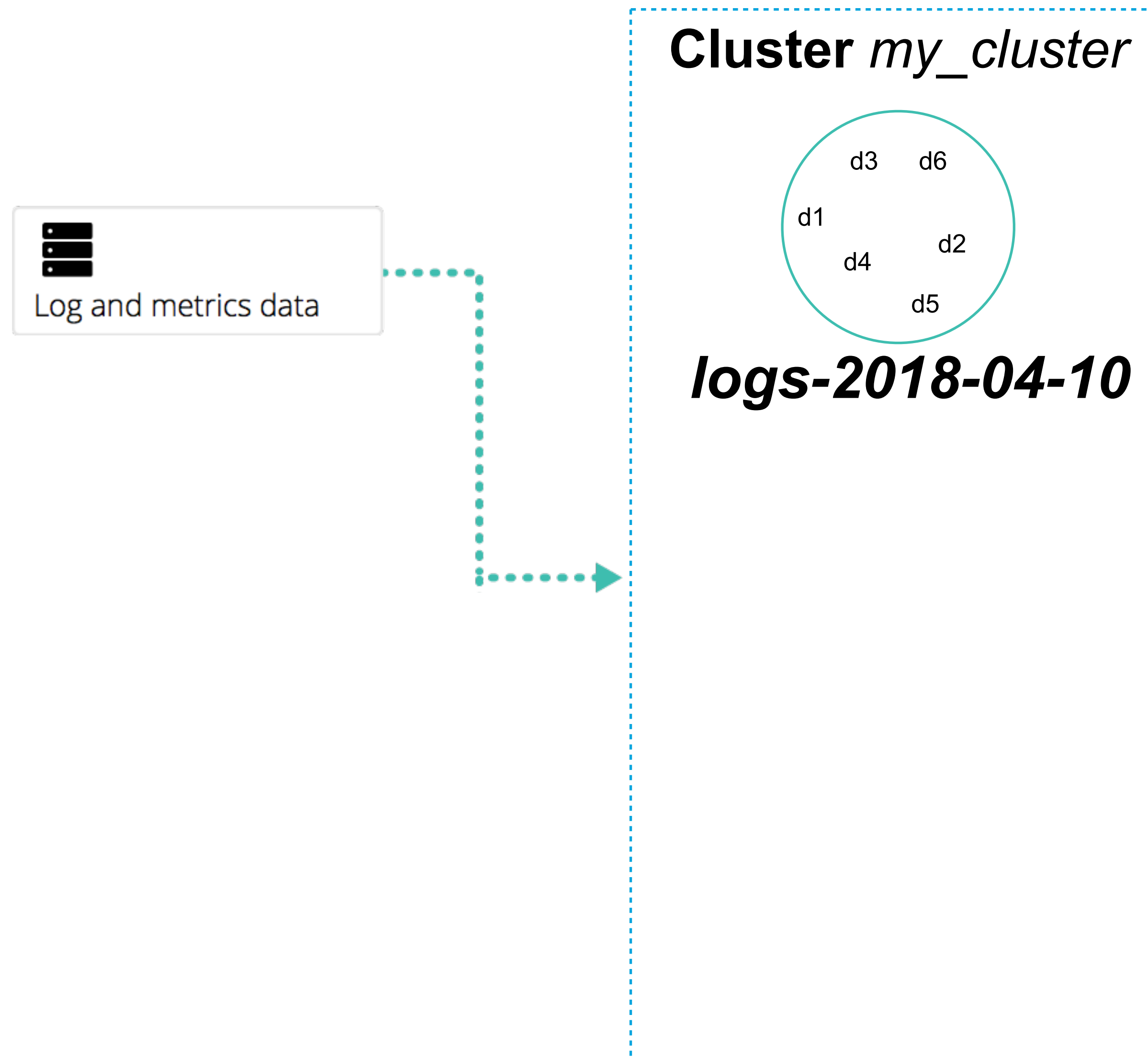
Time-Based Data

- Logs, social media streams, time-based events
- Timestamp + Data
- Do not change
- Typically search for recent events
- Older documents become less important
- Hard to predict the data size

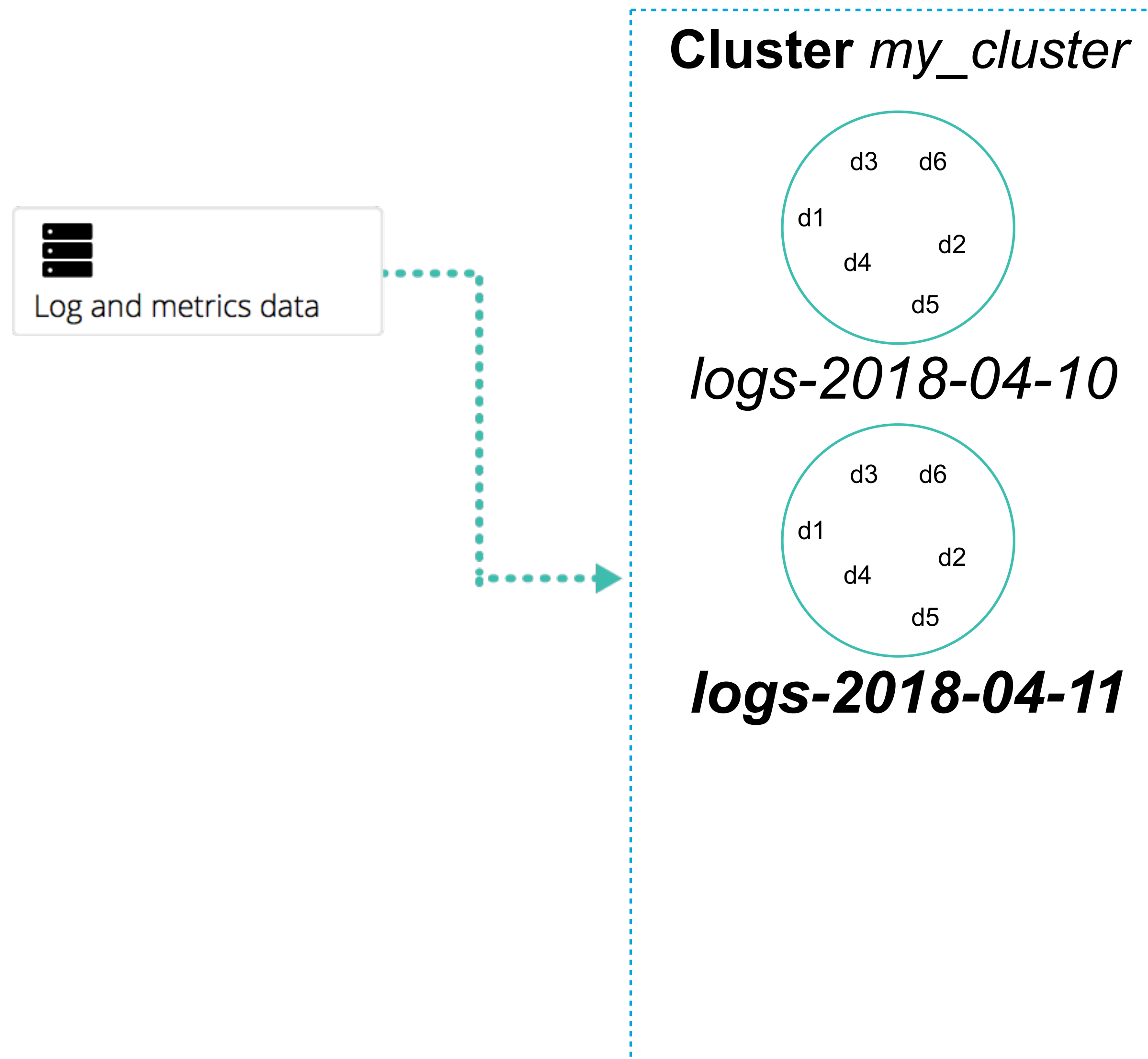
Time-Based Data

- Time-based Indices is the best option
 - create a new index each day, week, month, year, ...
 - search the indices you need in the same request

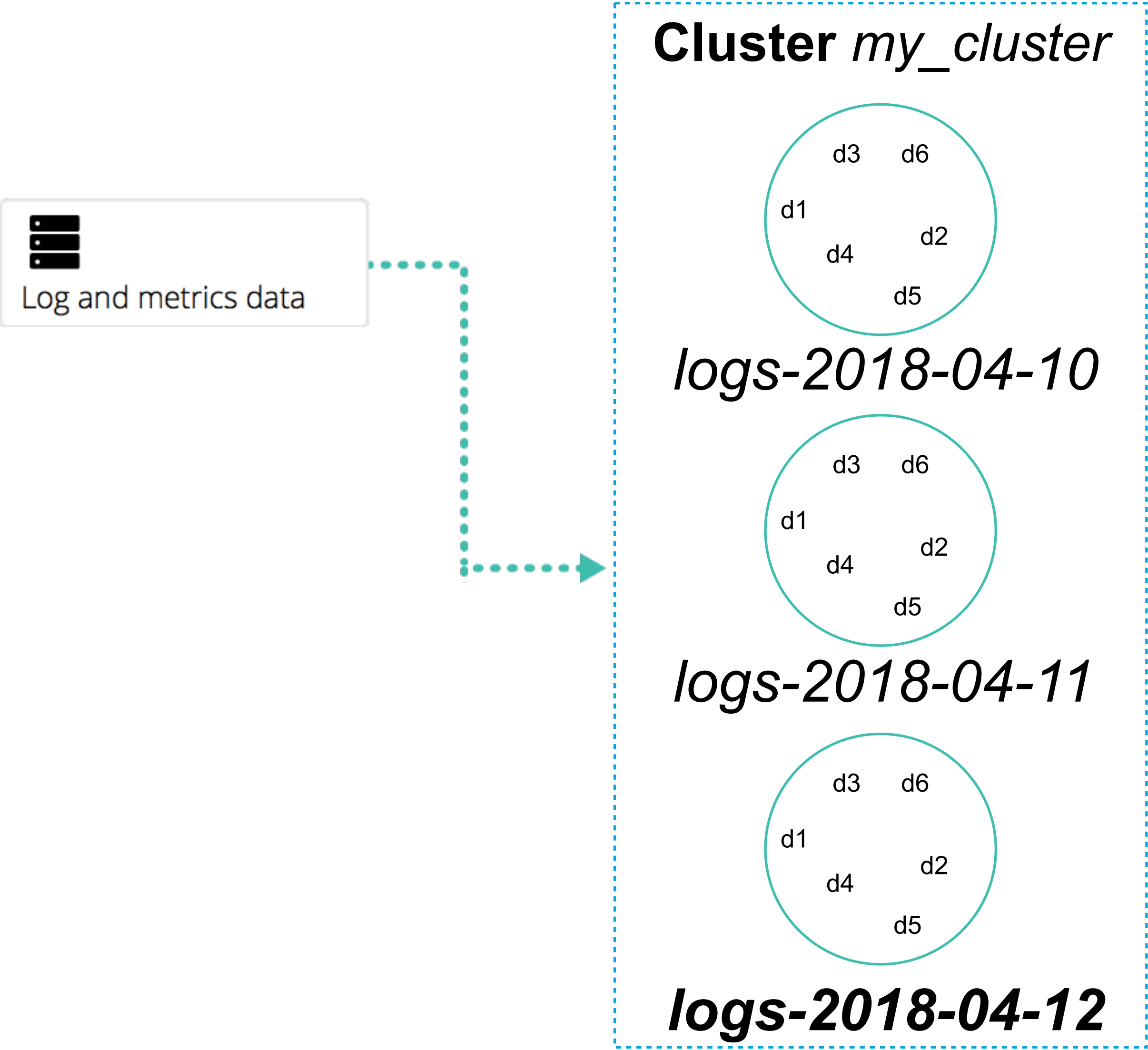
Daily Indices



Daily Indices



Daily Indices



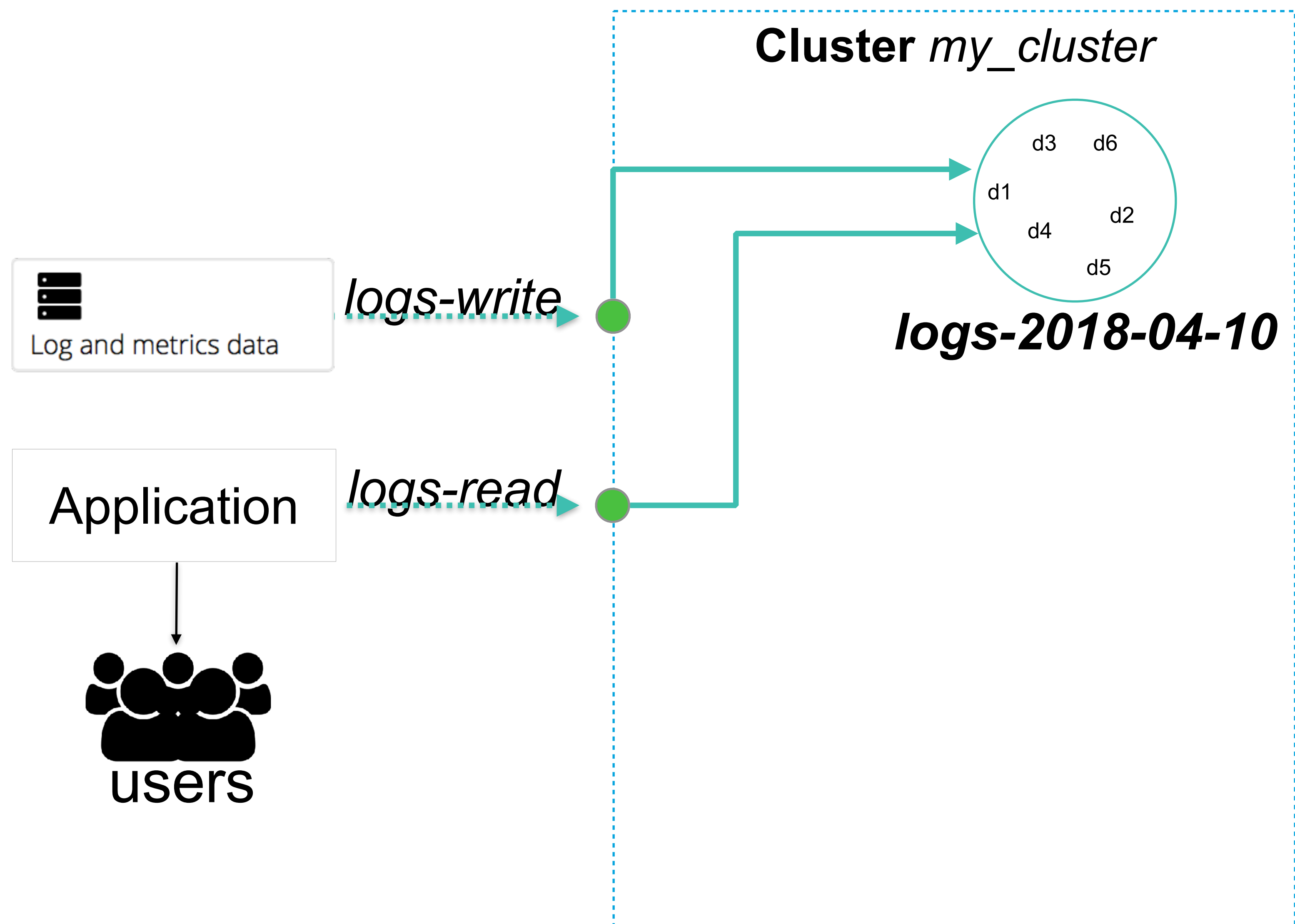
Templates

- Every new created index starting with 'logs-' will have
 - 2 shards
 - 1 replica (for each primary shard)
 - 60 seconds refresh interval

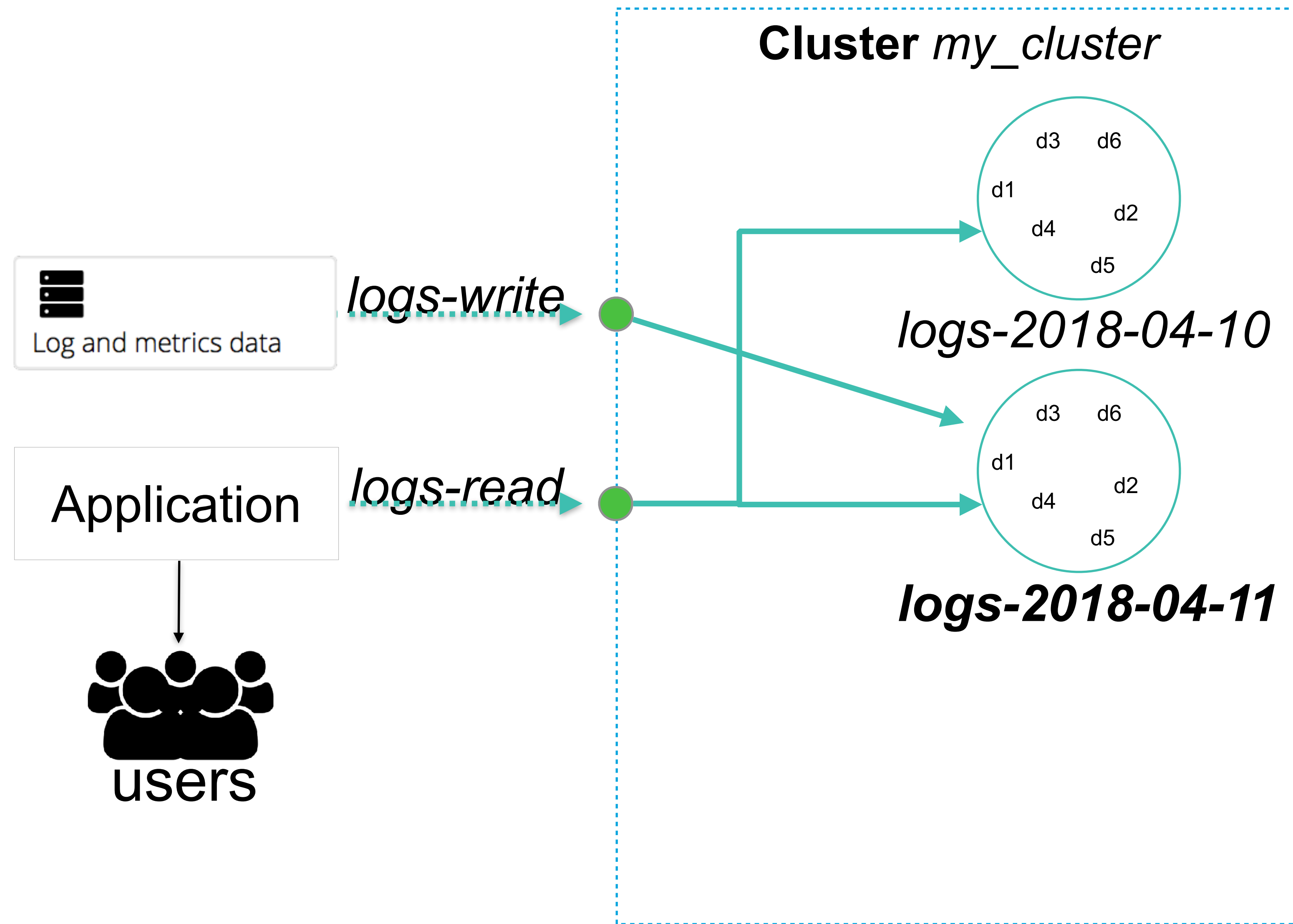
```
PUT _template/logs
{
  "template": "logs-*",
  "settings": {
    "number_of_shards": 2,
    "number_of_replicas": 1,
    "refresh_interval": "60s"
  }
}
```

More on that later

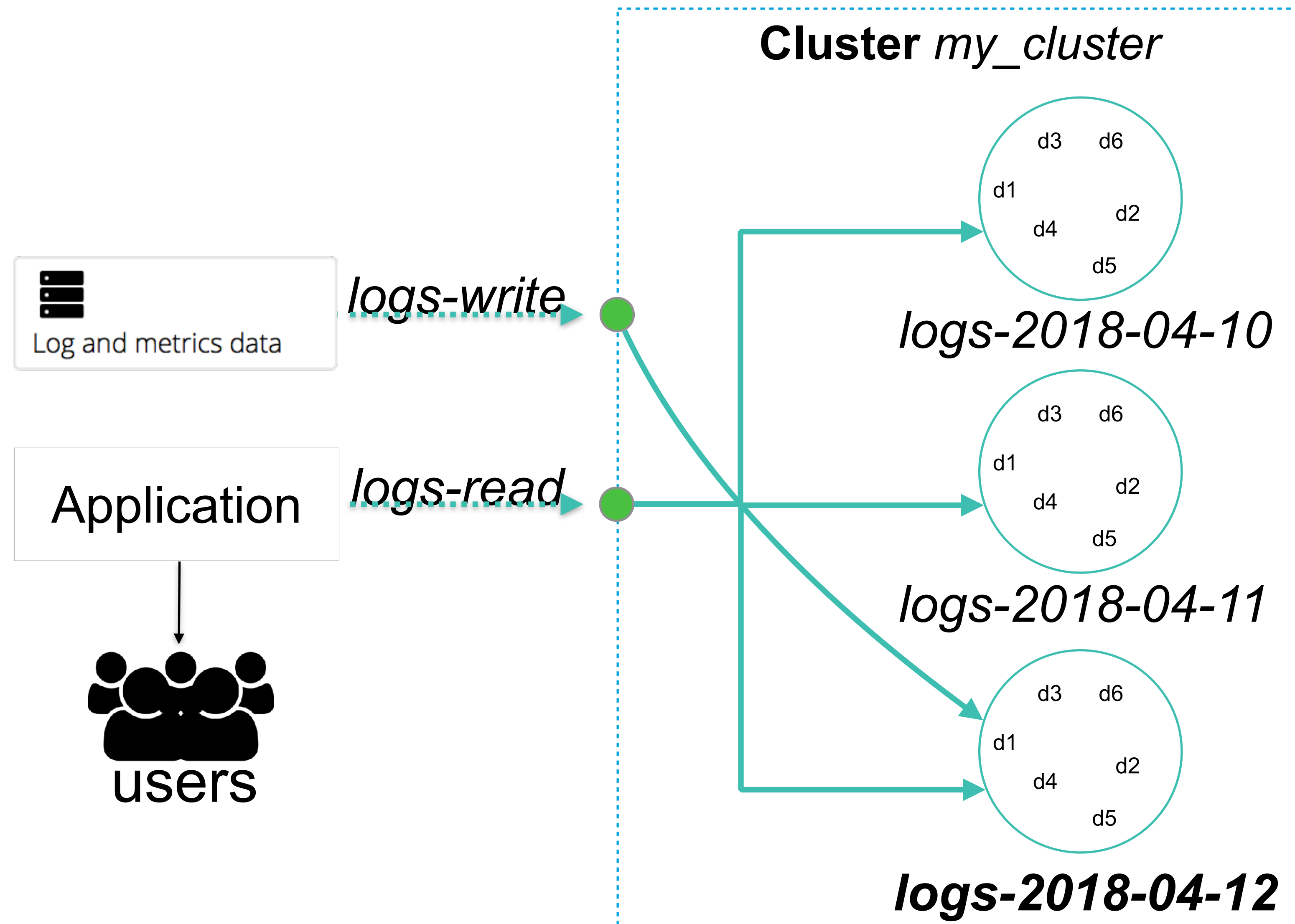
Alias



Alias



Alias





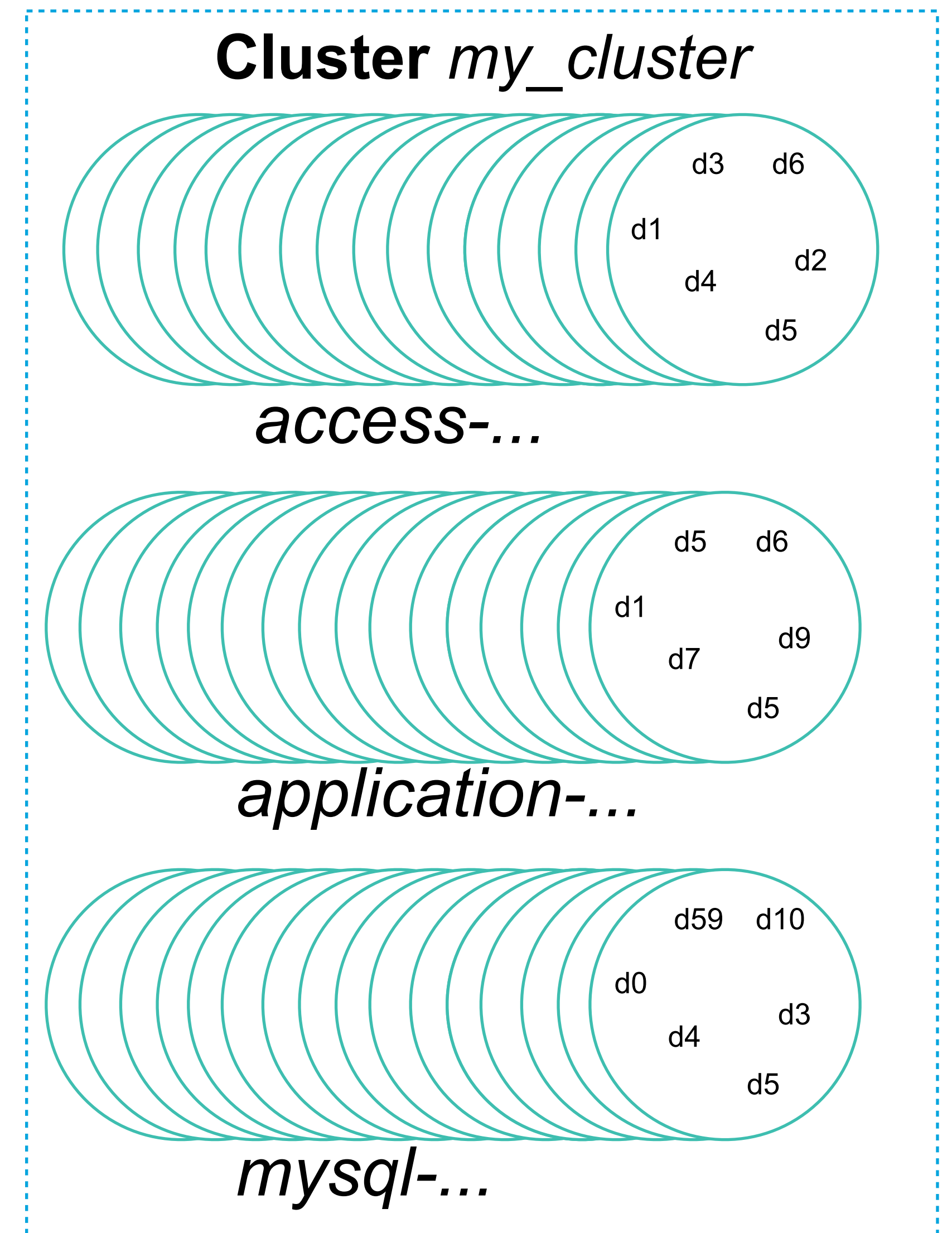
Detour: Rollover API

<https://www.elastic.co/guide/en/elasticsearch/reference/6.3/indices-rollover-index.html>

Do not Overshard

don't keep default values!

- 3 different logs
- 1 index per day each
- 1GB each
- 5 shards (default): so 200mb / shard vs 45gb
- 6 months retention
- **~900 shards for ~180GB**
- **we needed ~4 shards!**





Too many opened files



Elasticsearch

We have below cluster configuration:

1. There are 3 ES nodes with 16 GB memory on each server and 9 GB allocated to ES service on each node. One is master+Data, other 2 nodes are only data nodes.
2. We have time based indexing which creates indexes everyday. Example, system1-2017-04-03 and system1-2017-04-04, system1-2017-04-05 and so on.....
3. We have around 20 such systems. Logs from those many systems are stored in new index, every day.
4. We have 3 replicas and 3 shards settings for every index.
5.

```
cat /proc/sys/fs/file-max
512000
ulimit -n
65535
```


Scaling



But what happens if we have 2M users?

Scaling



Scaling



Scaling



Shards are the working unit

- Primaries
 - More data -> More shards
 - **write throughput** (More writes -> **More primary shards**)
- Replicas
 - high availability (1 replica is the default)
 - **read throughput** (More reads -> **More replicas**)



Detour: Shrink API

<https://www.elastic.co/guide/en/elasticsearch/reference/6.3/indices-shrink-index.html>



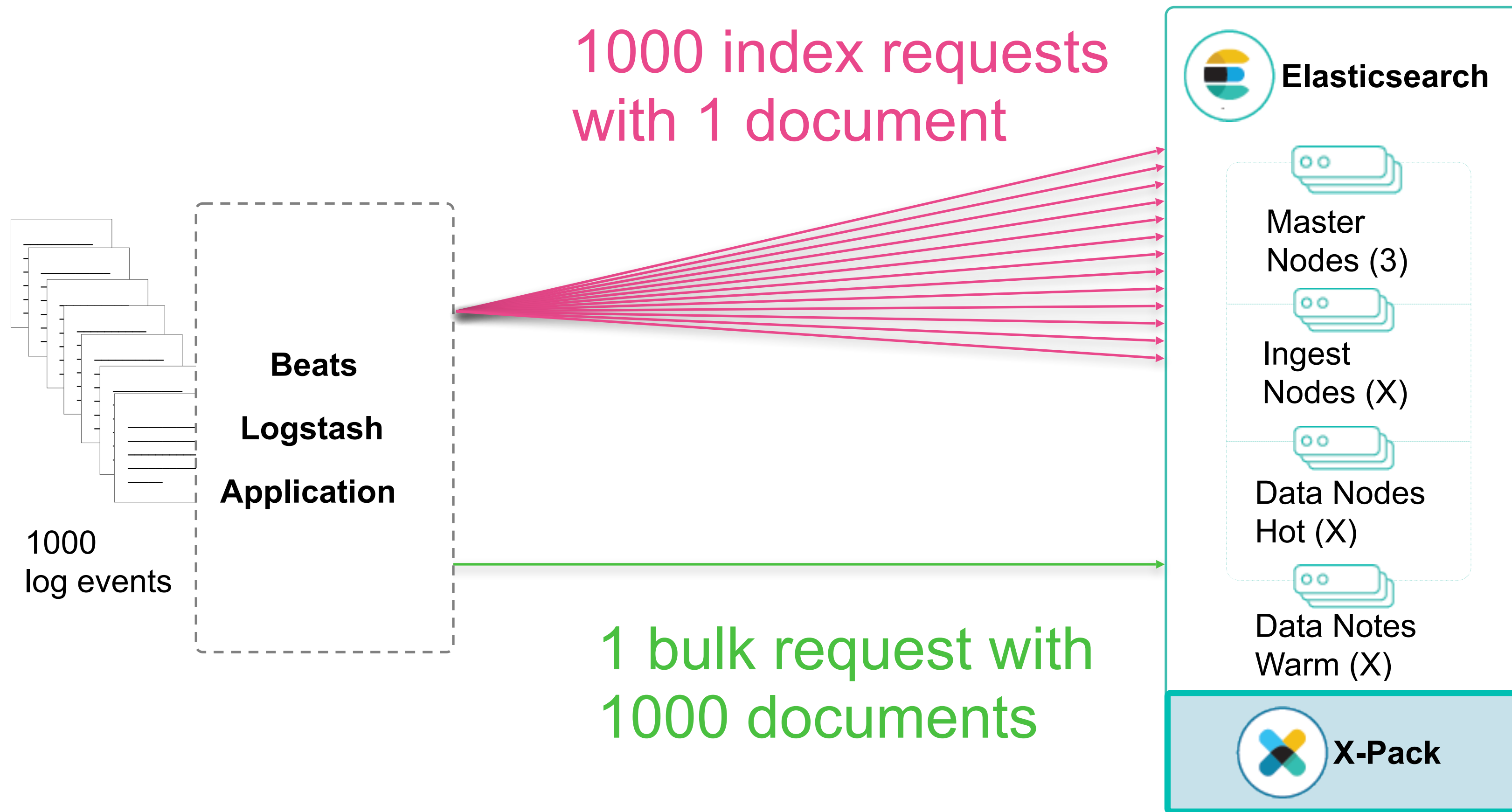
Detour: Split API

<https://www.elastic.co/guide/en/elasticsearch/reference/6.3/indices-split-index.html>

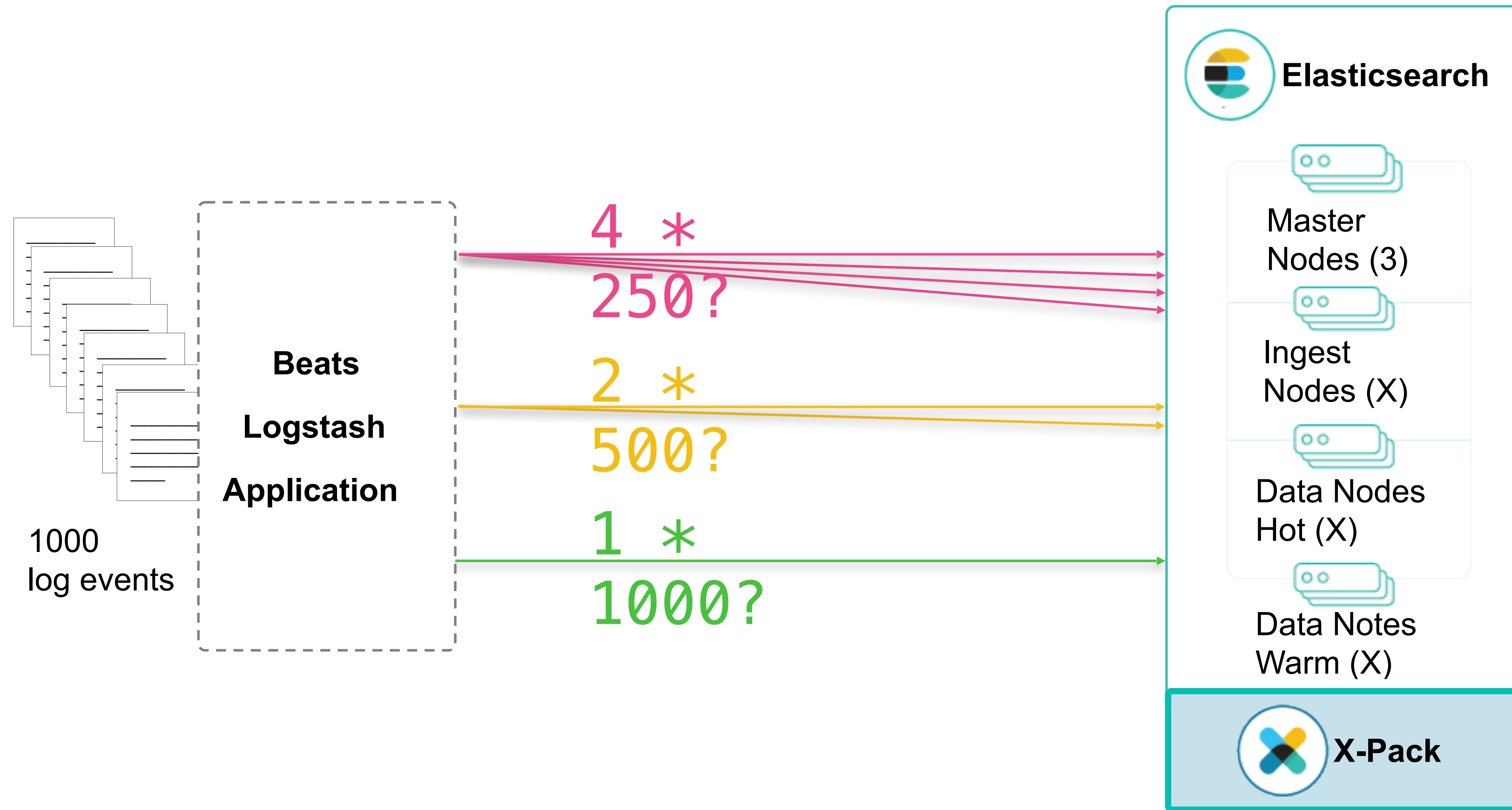


Optimal Bulk Size

What is Bulk?



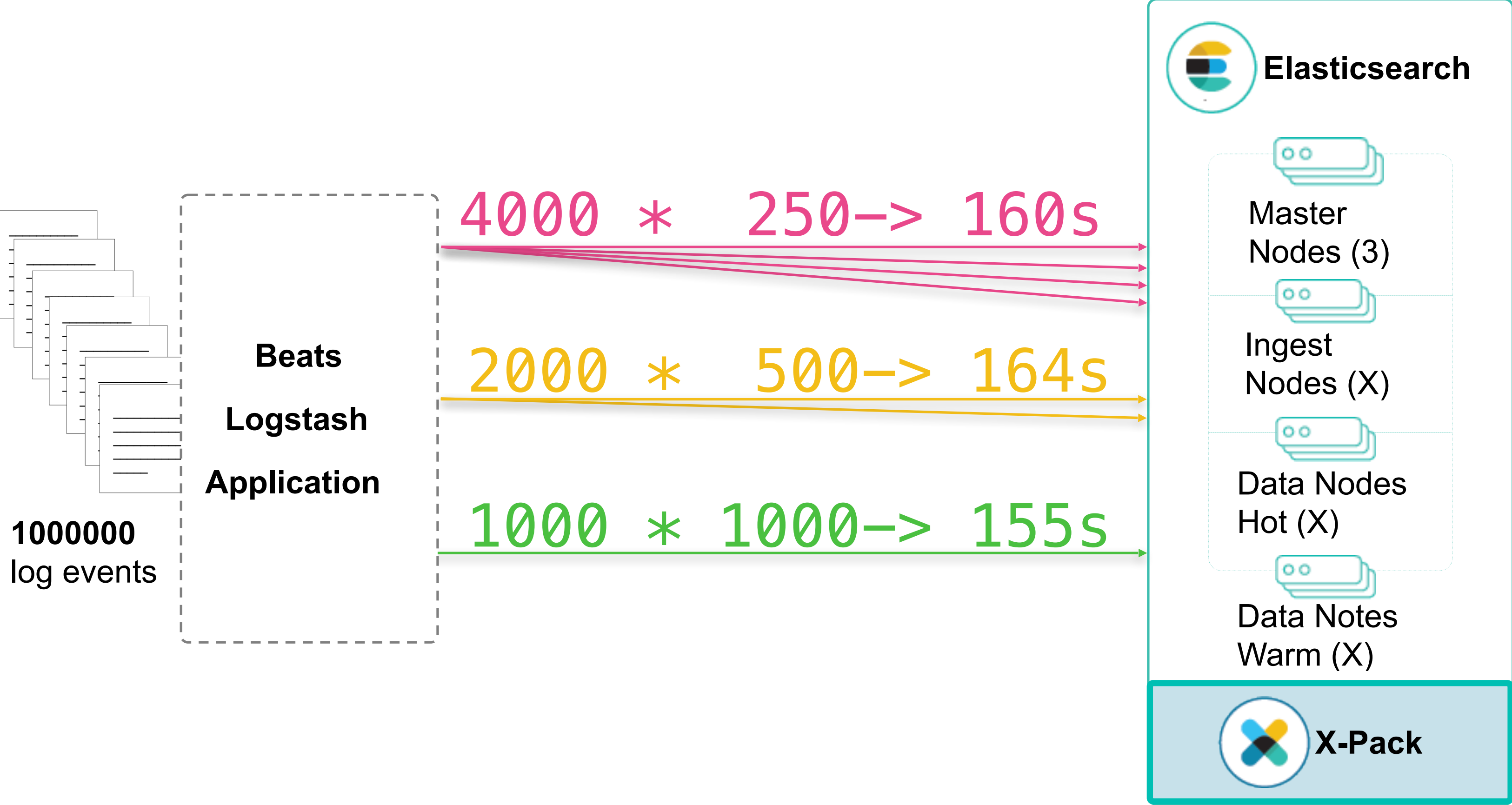
What is the optimal bulk size?



It depends...

- on your application (language, libraries, ...)
- document size (100b, 1kb, 100kb, 1mb, ...)
- number of nodes
- node size
- number of shards
- shards distribution

Test it ;)



Test it ;)

```
input { stdin{} }

filter {}

output {
  elasticsearch {
    hosts => ["10.12.145.189"]
    flush_size => "${SIZE}"
  }
}
```

In Beats set "**bulk_max_size**"
in the **output.elasticsearch**

```
DATE=`date +%Y.%m.%d`
LOG=logs/logs.txt

exec_test () {
  curl -s -XDELETE "http://USER:PASS@HOST:9200/logstash-$DATE"
  sleep 10
  export SIZE=$1
  time cat $LOG | ./bin/logstash -f logstash.conf
}

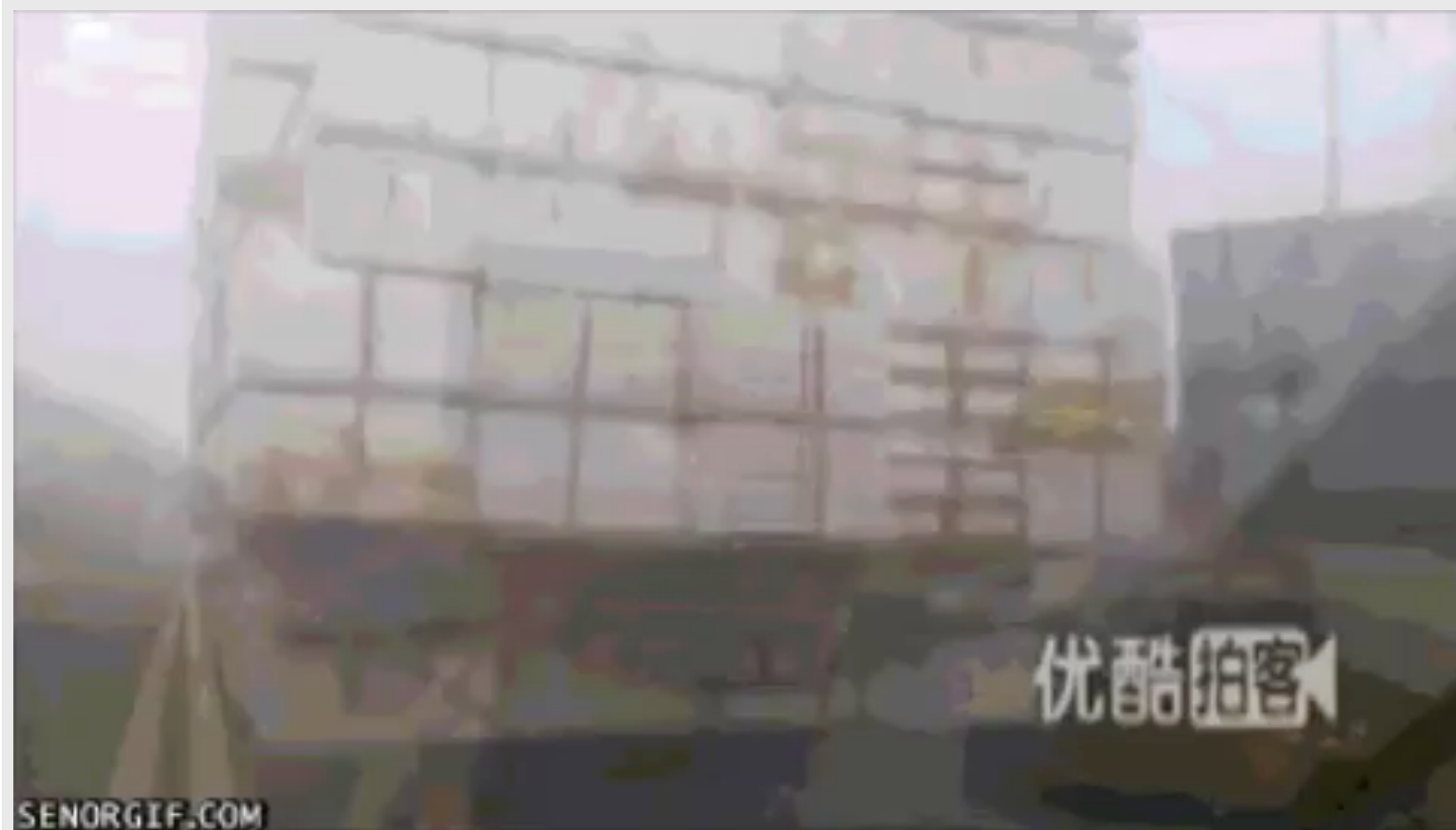
for SIZE in 100 500 1000 3000 5000 10000; do
  for i in {1..20}; do
    exec_test $SIZE
  done; done;
```

Test it ;)

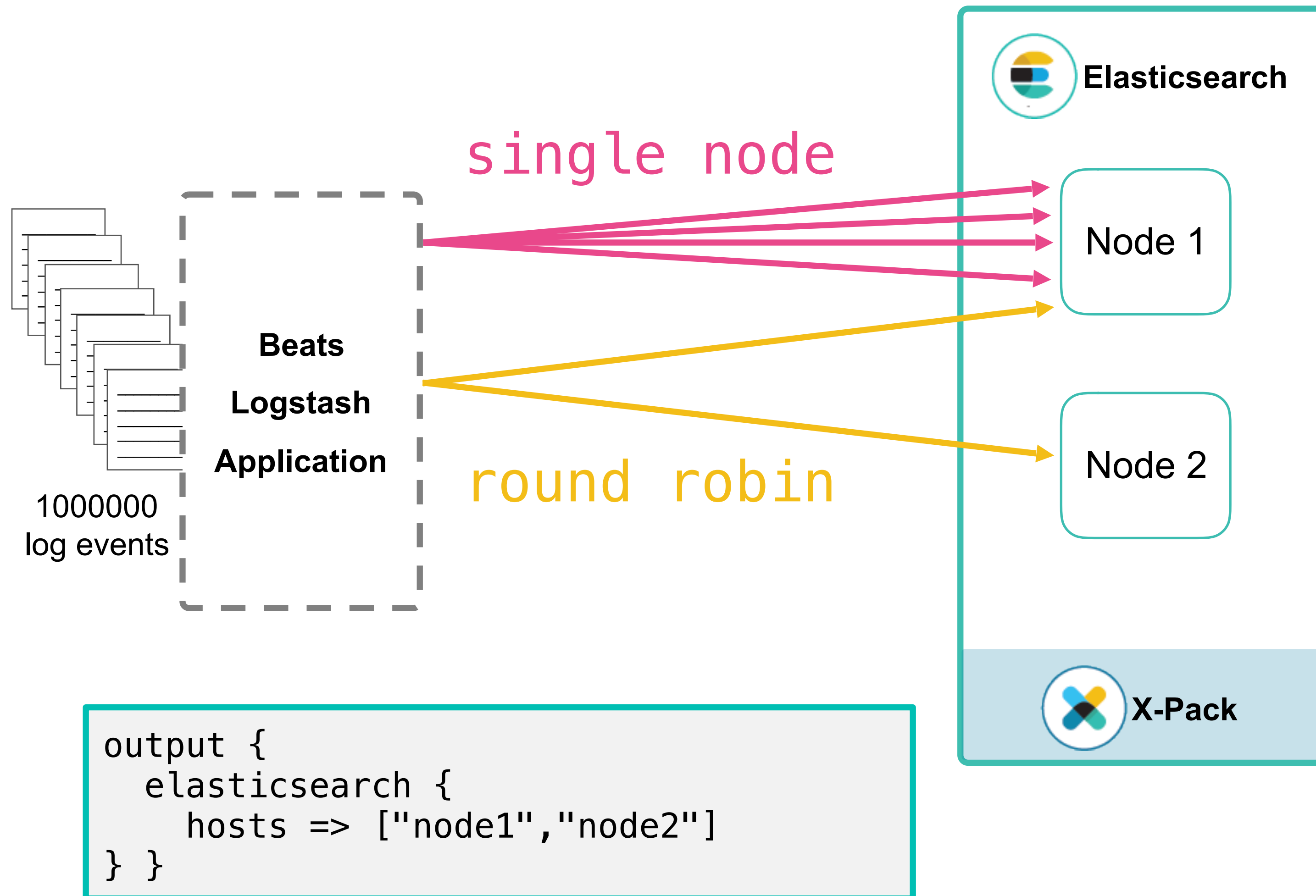
- 2 node cluster (m3.large)
 - 2 vCPU, 7.5GB Memory, 1x32GB SSD
- 1 index server (m3.large)
 - logstash
 - kibana

# docs	100	500	1000	3000	5000	10000
time(s)	191.7	161.9	163.5	160.7	160.7	161.5

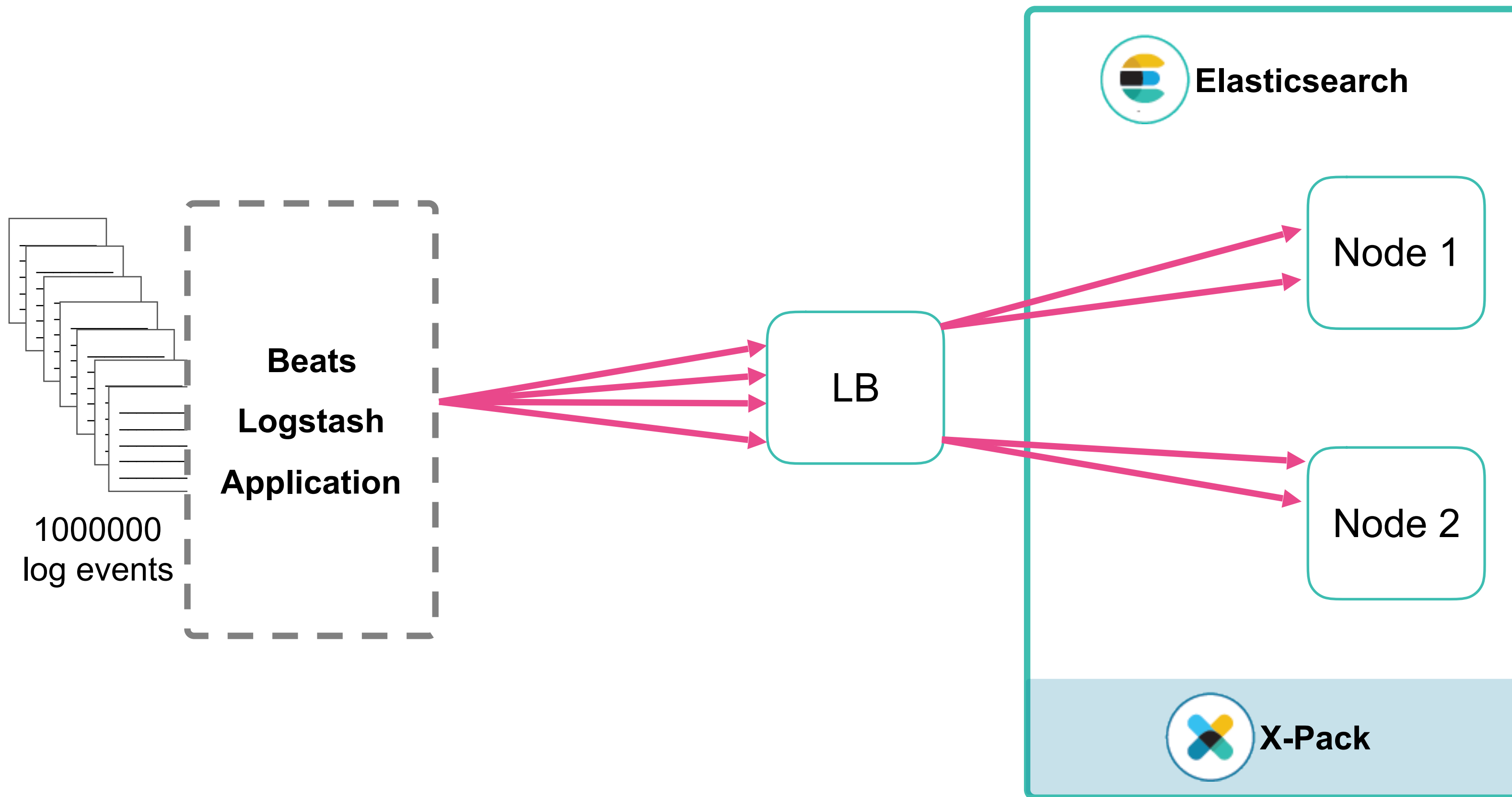
Distribute the Load



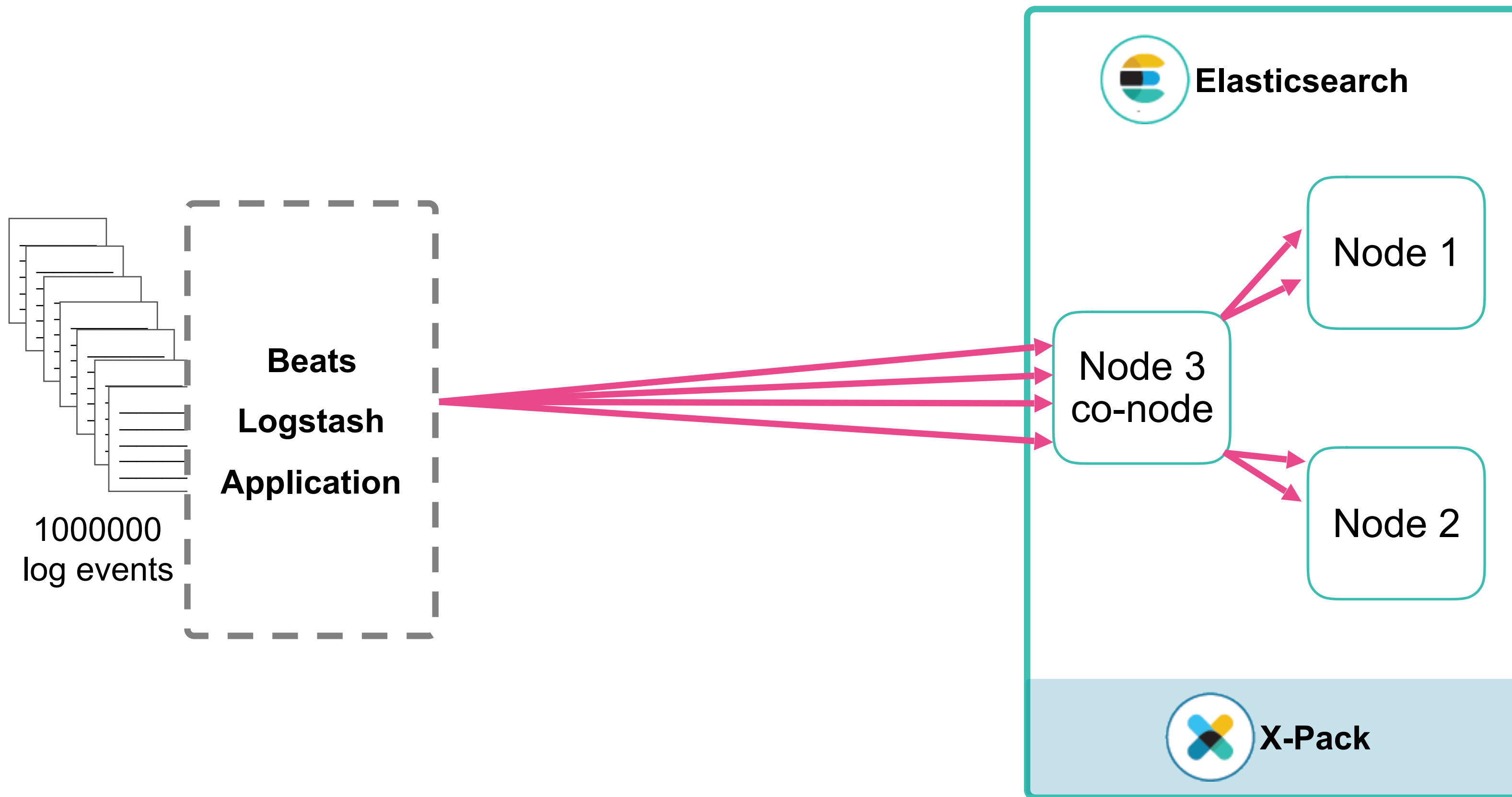
Avoid Bottlenecks



Load Balancer



Coordinating-only Node



Test it ;)

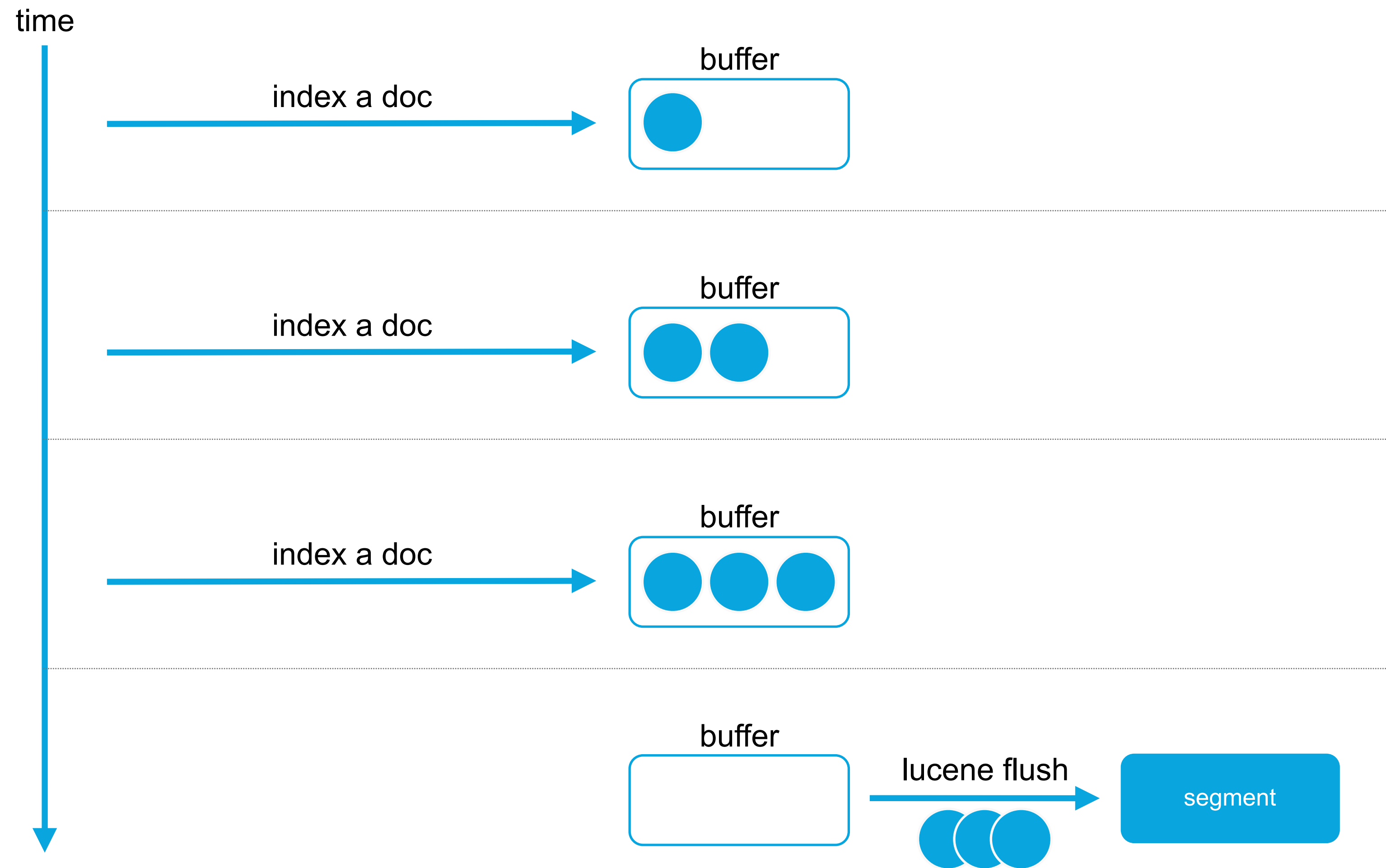
- 2 node cluster (m3.large)
 - 2 vCPU, 7.5GB Memory, 1x32GB SSD
- 1 index server (m3.large)
 - logstash (round robin configured)
 - `hosts => ["10.12.145.189", "10.121.140.167"]`
 - kibana

#docs time(s)	1000	5000	10000
NO Round Robin	163.5	160.7	161.5
Round Robin	161.3	158.2	159.4



Optimizing Disk IO

Durability



refresh_interval

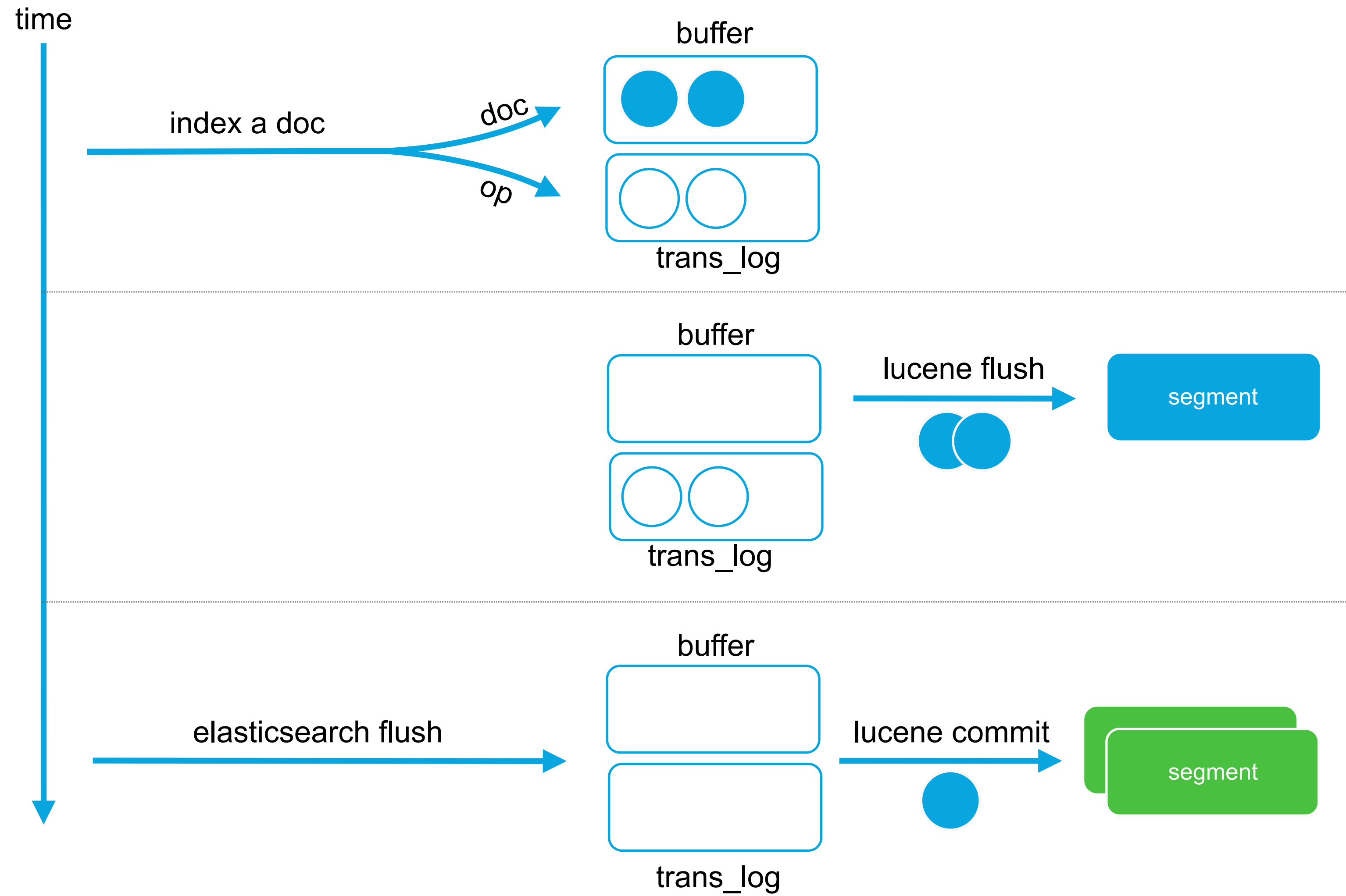
- Dynamic per-index setting

```
PUT logstash-2017.05.16/_settings
{
  "refresh_interval": "60s"
}
```

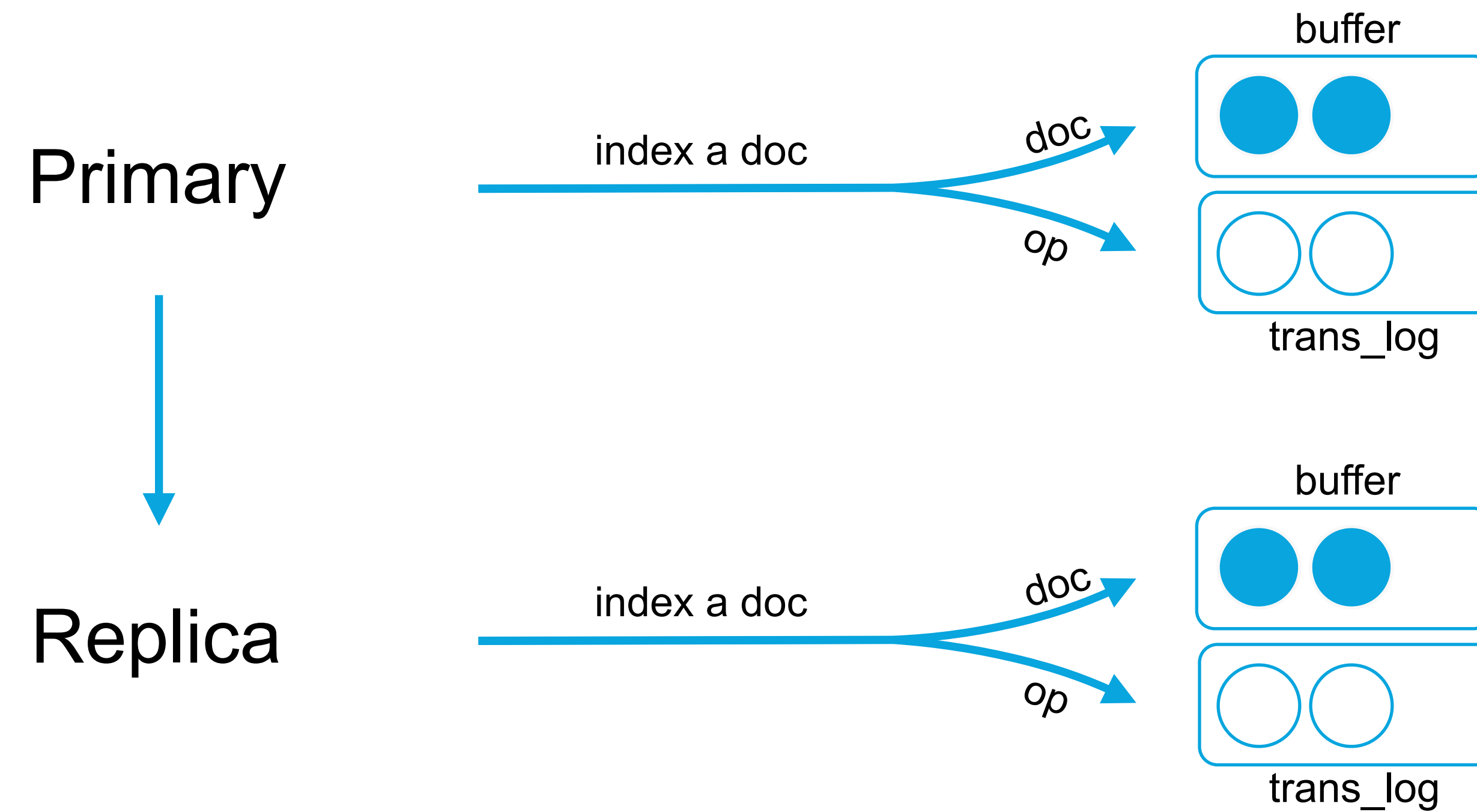
- Increase to get better write throughput to an index
- New documents will take more time to be available for Search.

#docs time(s)	1000	5000	10000
1s refresh	161.3	158.2	159.4
60s refresh	156.7	152.1	152.6

Durability



Translog fsync every 5s (1.7)



redundancy doesn't help if all nodes lose power

Async Transaction Log

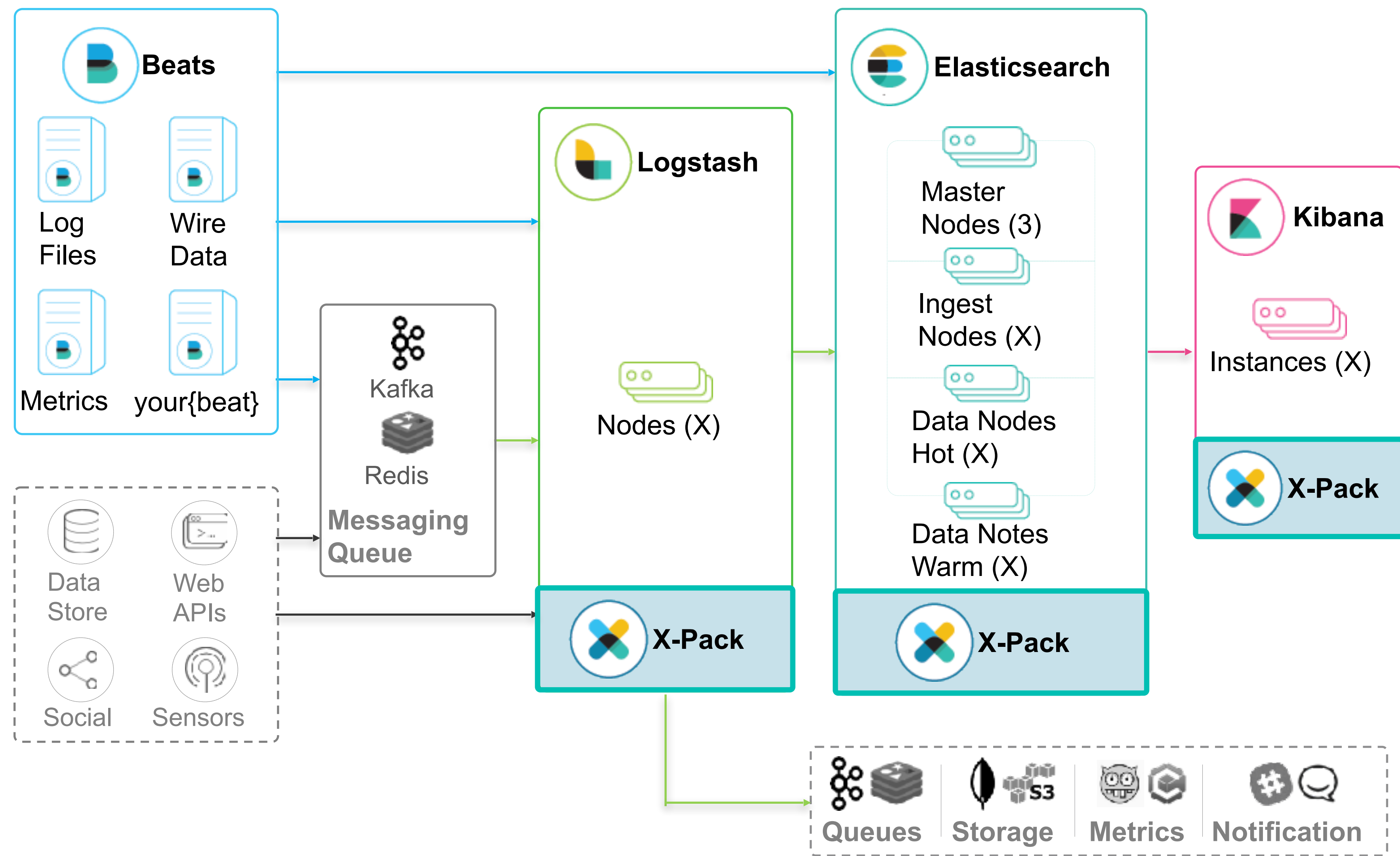
- `index.translog.durability`
 - request (default)
 - async
- `index.translog.sync_interval` (only if async is set)
- Dynamic per-index settings
- **Be careful, you are relaxing the safety guarantees**

#docs time(s)	1000	5000	10000
Request fsync	161.3	158.2	159.4
5s sync	152.4	149.1	150.3



Final Remarks

Final Remarks



Final Remarks

- Primaries
 - More data -> More shards
 - Do not overshard!
- Replicas
 - high availability (1 replica is the default)
 - read throughput (More reads -> More replicas)



Final Remarks

- Bulk and Test
- Distribute the Load
- Refresh Interval
- Async Trans Log (**careful**)

#docs per bulk	1000	5000	10000
Default	163.5	160.7	161.5
RR+60s+Async5s	152.4	149.1	150.3



Les vendredi noirs ? Même pas peur !

Merci !

David Pilato
Developer | Evangelist, @dadoonet

VOXXEDDAYS
LUXEMBOURG

