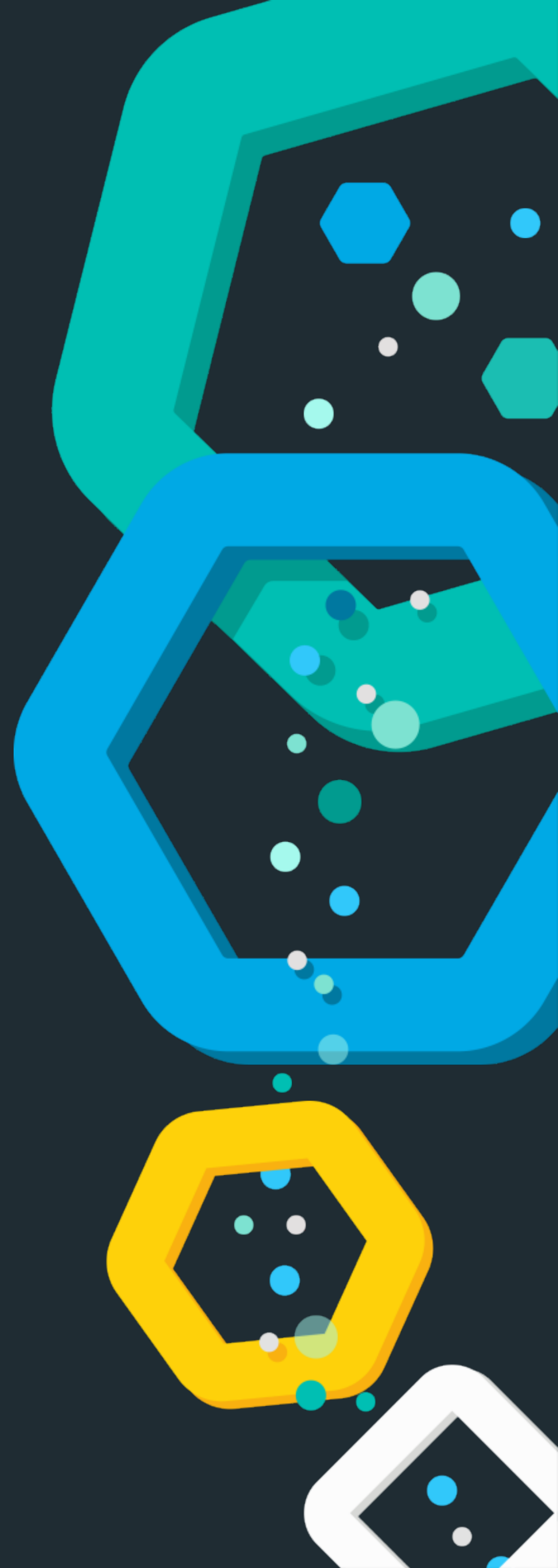


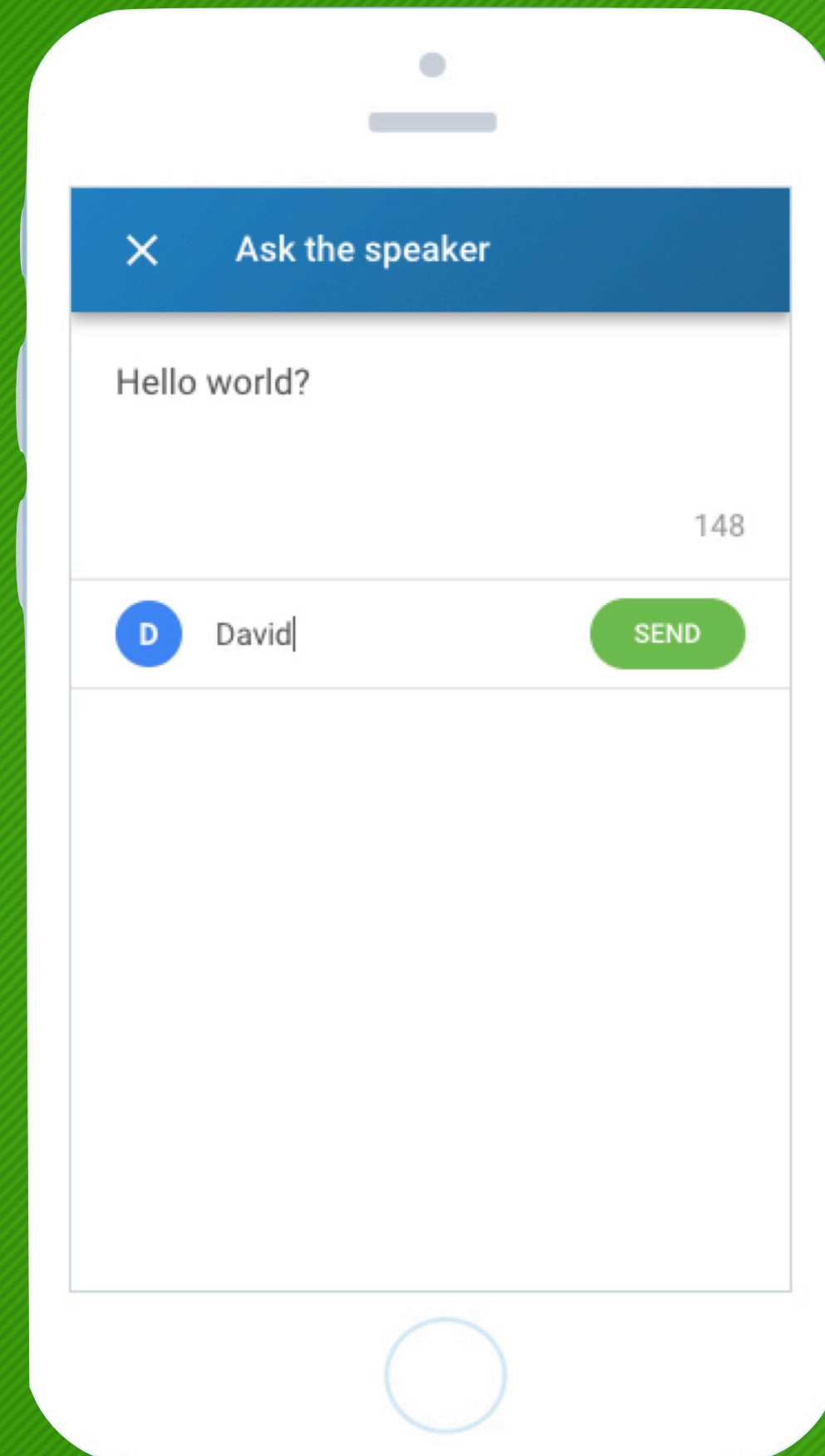


Déployer et monitorer avec elastic sur GCP

David Pilato
Developer | Evangelist, @dadoonet



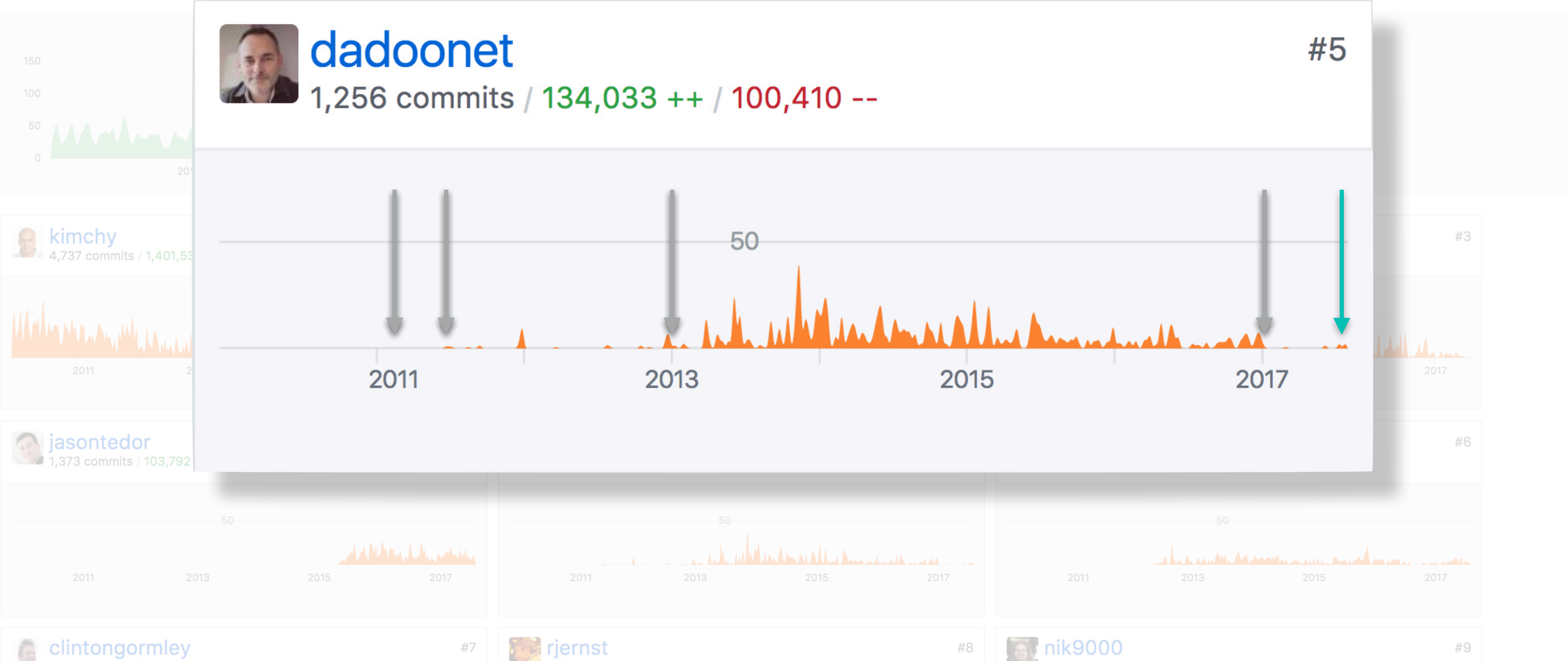
slido



Join at
slido.com
#elastic

Feb 7, 2010 – Oct 2, 2017
Contributions to master, excluding merge commits

Contributions: Commits



Agenda

- 1 Data Platform Architectures
- 2 Deploying on Google platform
- 3 Elasticsearch Cluster Sizing
- 4 Optimal Bulk Size
- 5 Distribute the Load
- 6 Optimizing Disk IO
- 7 Final Remarks



Data Platform Architectures

fridge:leftovers

user:soulmate

life:universe

_Search? outside the box

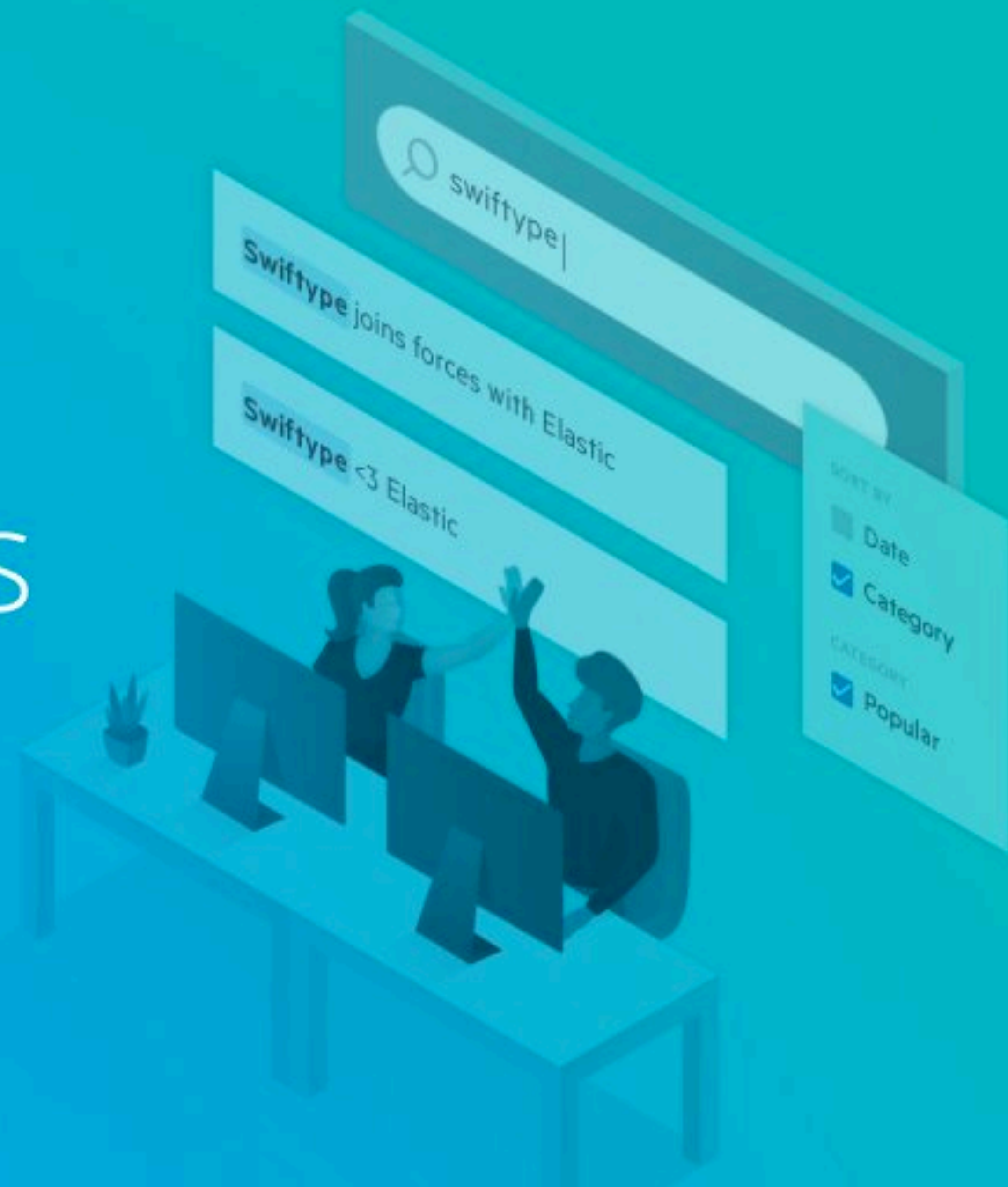
city:restaurant

car:model

work:dreamjob



Elastic Joins Forces with Swifttype

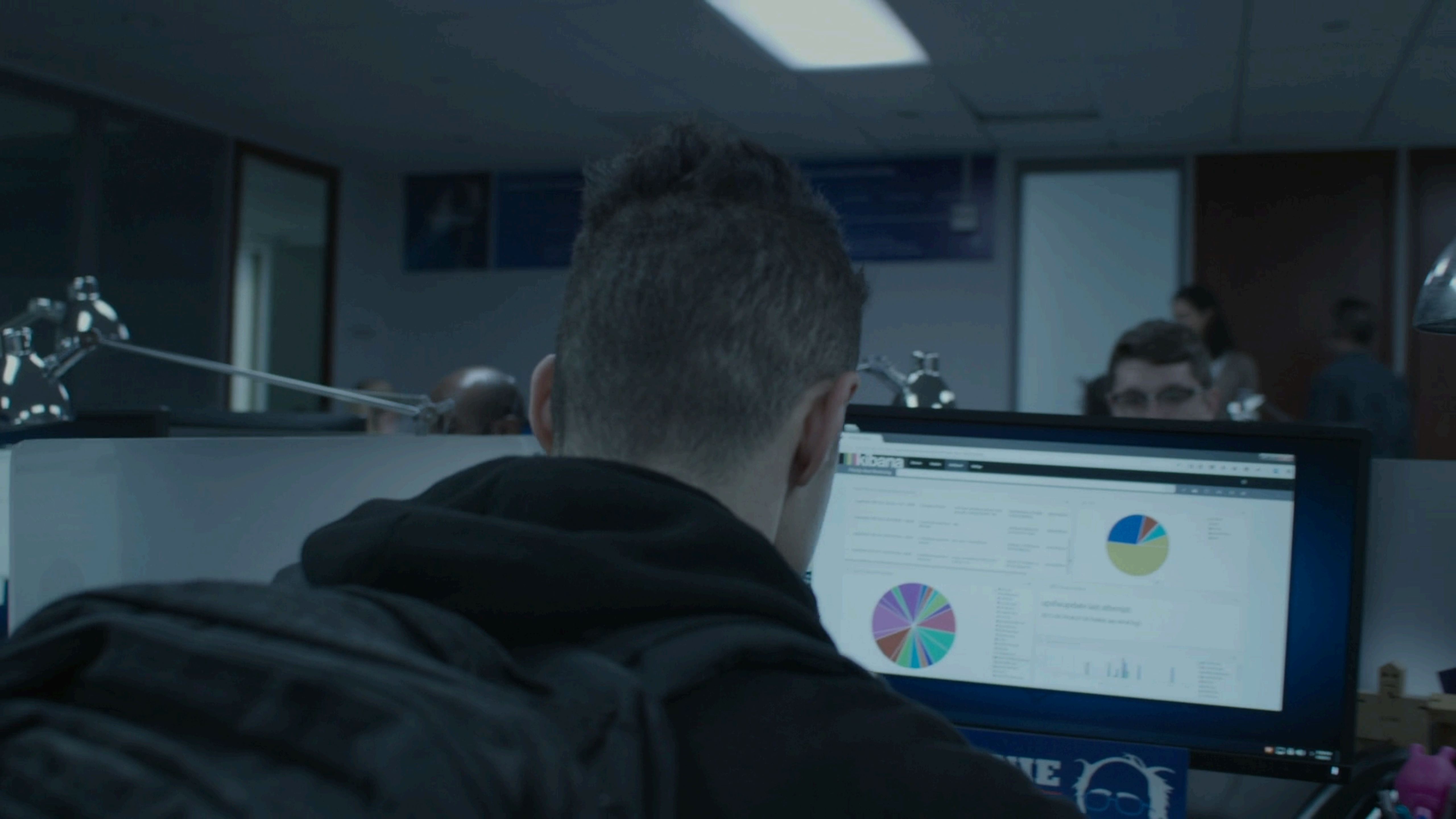


Logging

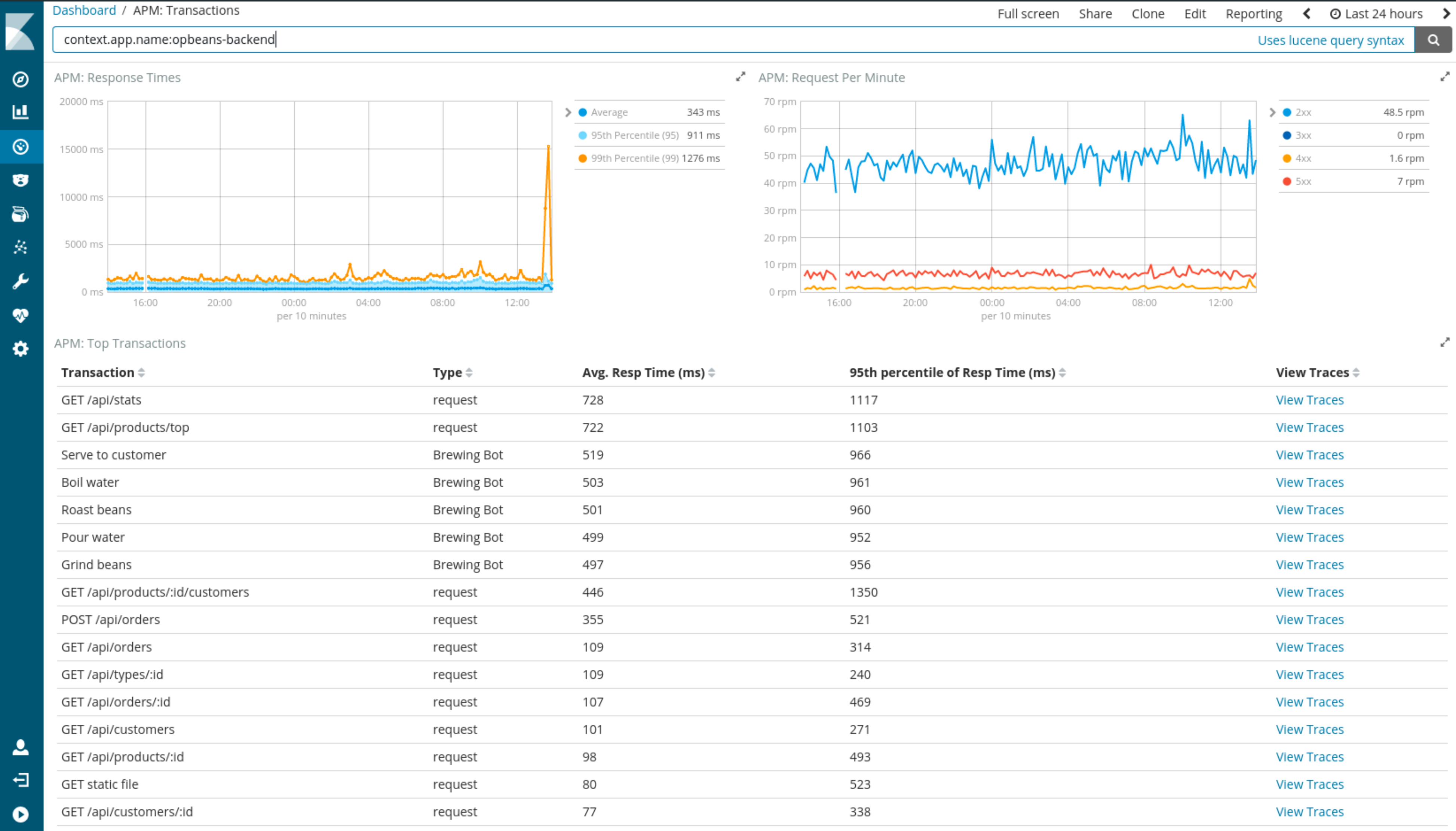


Metrics

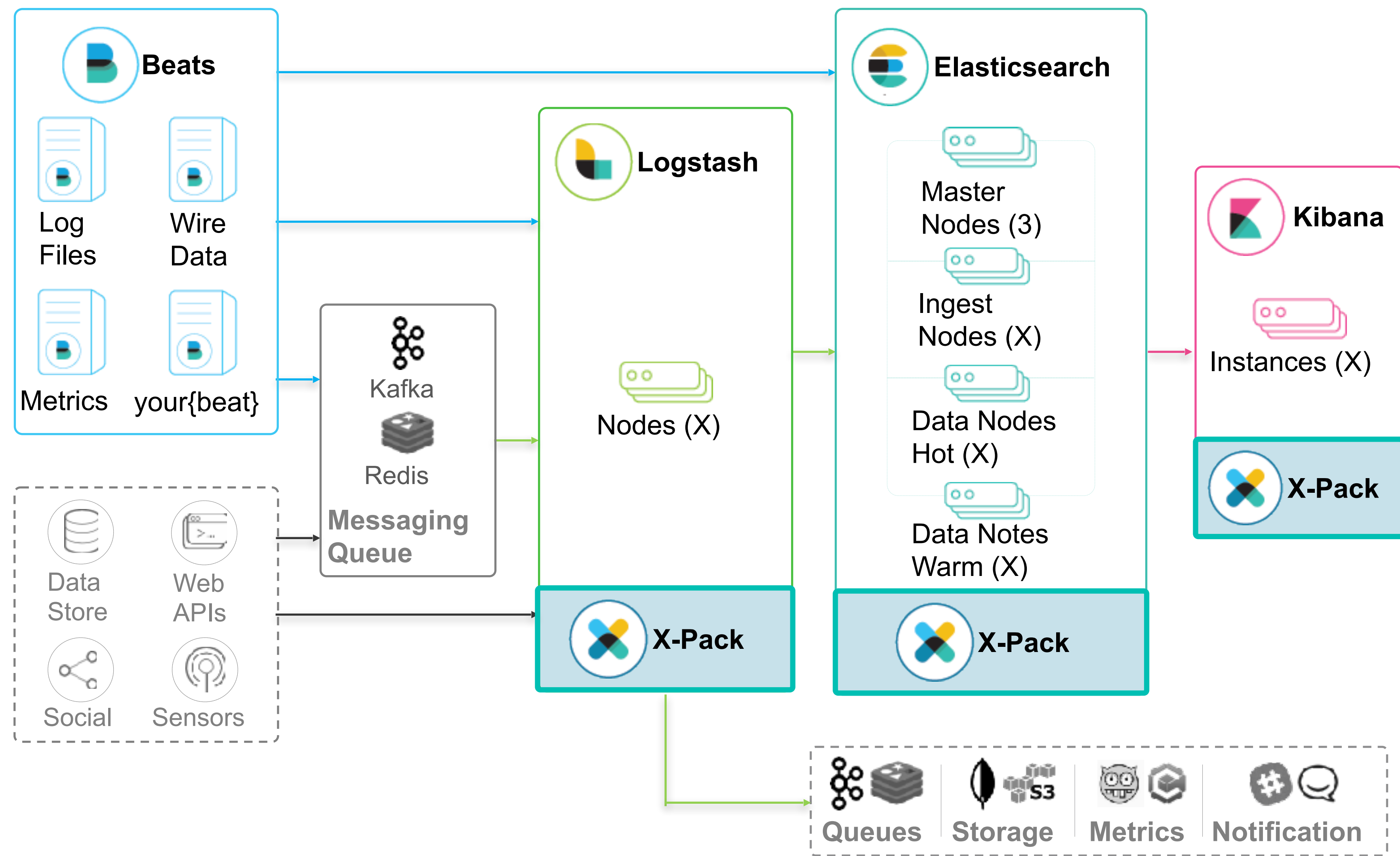




APM

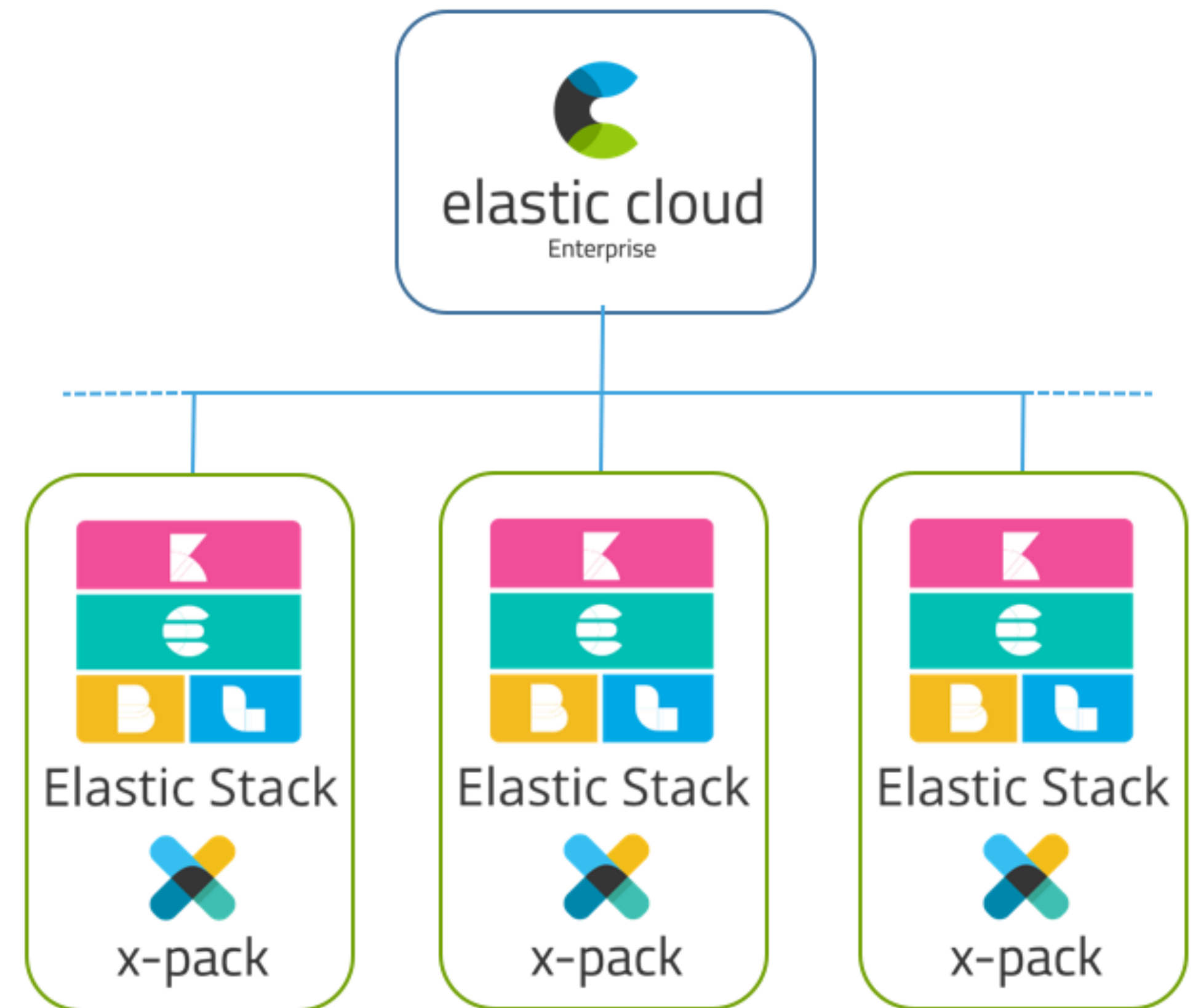


The Elastic Journey of Data





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

The screenshot shows the Elastic Cloud console interface. At the top, there's a navigation bar with 'cloud', 'Clusters', 'Plugins', 'Help', 'Account', and 'Sign out'. The main content is divided into two columns. The left column, titled 'Summary', contains a table with the following 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
Monthly rate	\$45

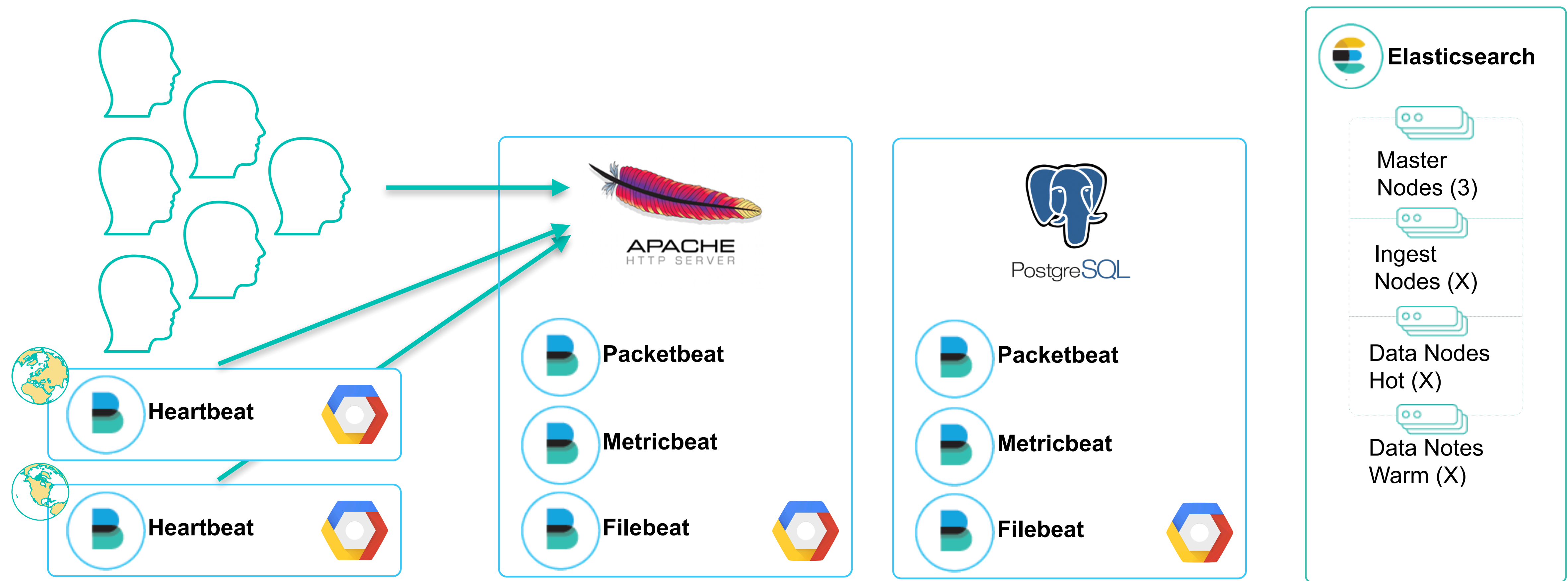
Below the table is a blue 'Create' button. The right column, titled 'Cluster Size', has a sub-header 'Choose a cluster size. Cluster size can be changed later without downtime.' It features a slider with memory and storage options ranging from 1GB/24GB to 256GB/6144GB. A 'Recommended for production' range is indicated between 4GB/96GB and 64GB/1536GB. Below the slider, there's a checkbox for 'SSD' which is checked, with the text 'Selected for improved storage performance.' and a link 'Need a larger cluster? Contact us.' The 'Cloud Platform' section below has a 'Pick your cloud:' section with buttons for 'amazon web services' (selected) and 'Google Cloud Platform'. The 'Choose a region near you:' section has buttons for 'US East (N. Virginia)' (selected), '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

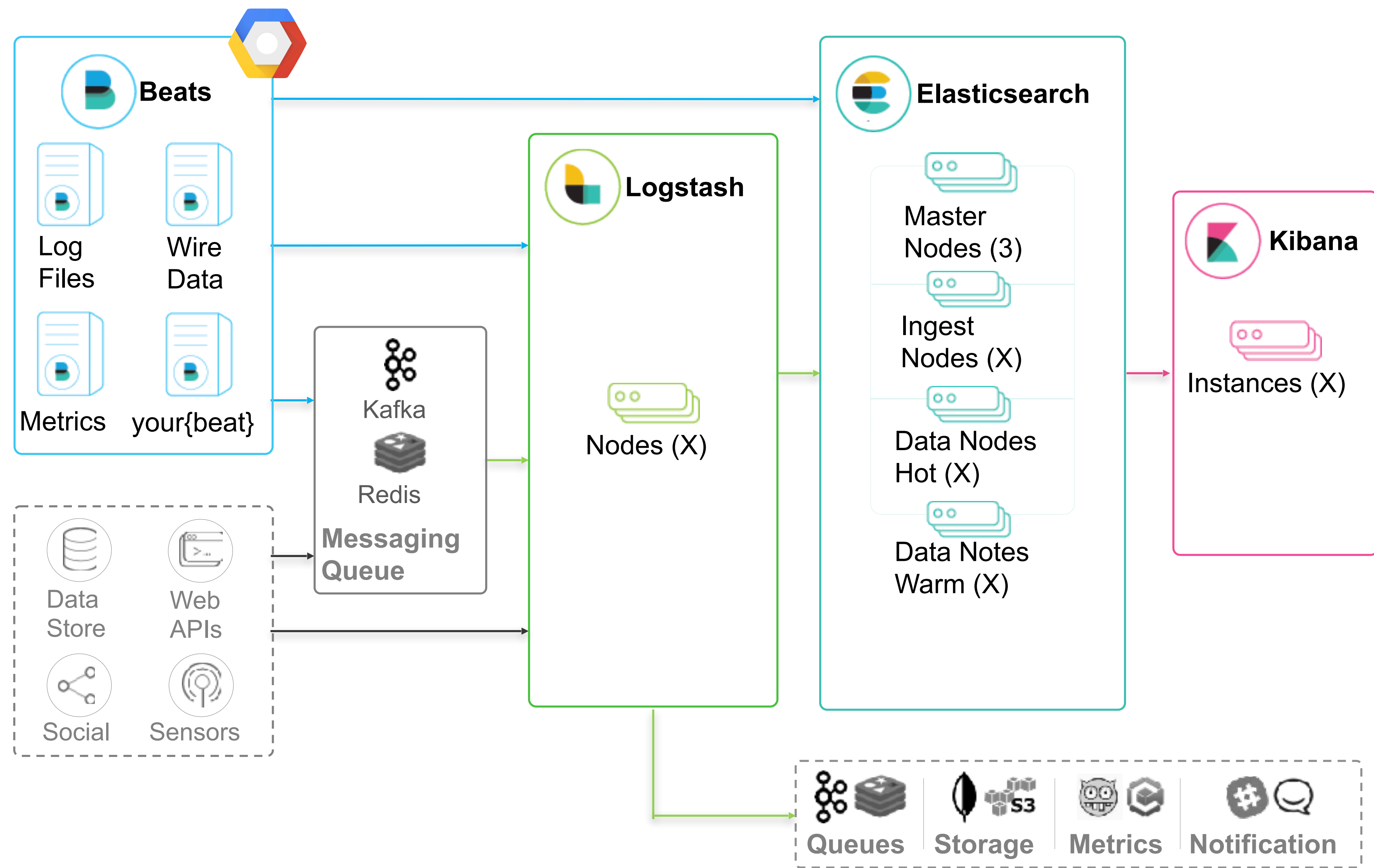


Deploying on Google Platform

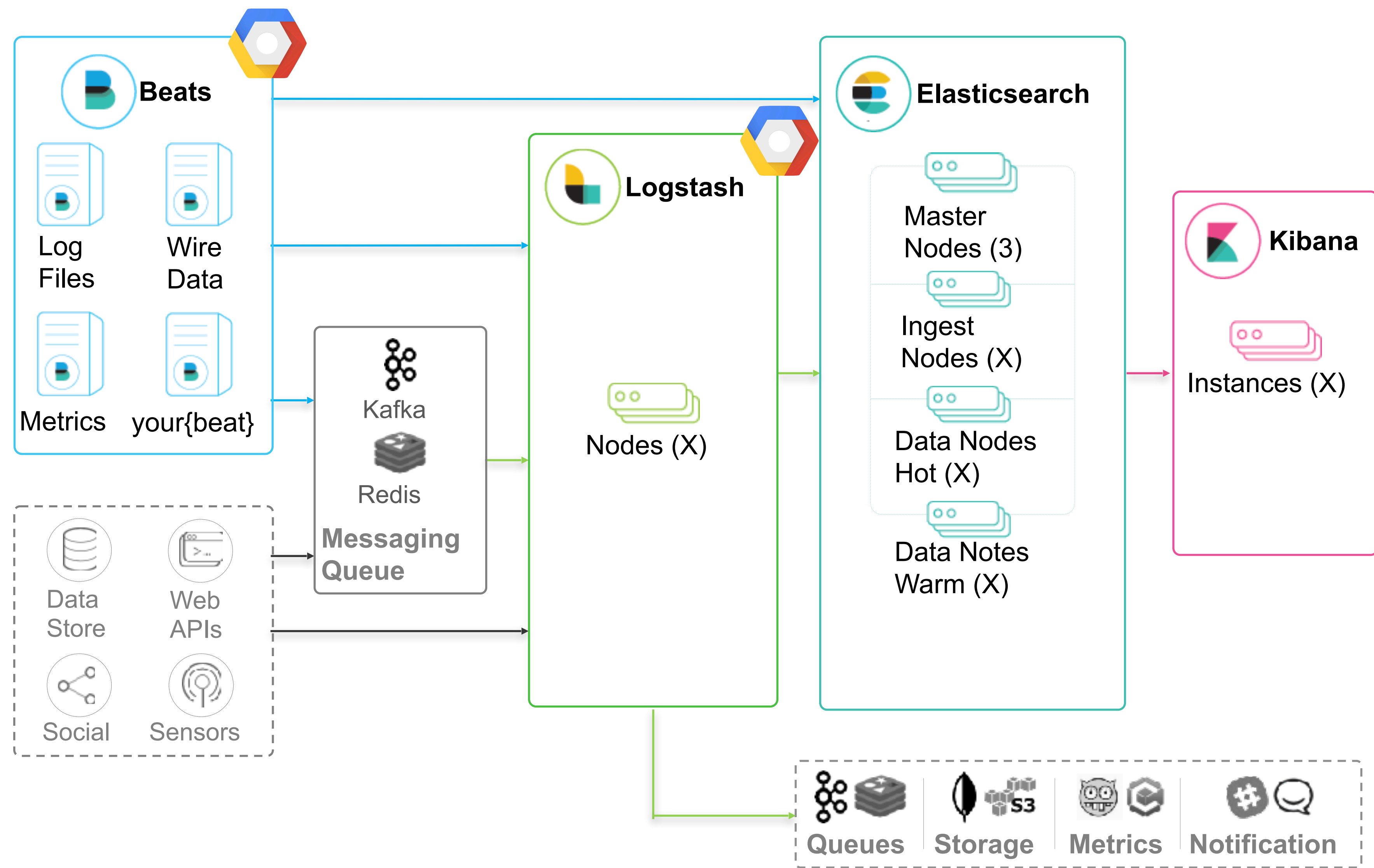
Deploying beats



Elastic Platform



Elastic Platform





elasticsearch

Elasticsearch Manual deployment

Manual deployment

```
$ gcloud compute instances create "dadoonet-1" --machine-type "n1-standard-1" \
  --scopes "https://www.googleapis.com/auth/cloud-platform" \
  --image "debian-9-stretch-v20171129" --image-project "debian-cloud" \
  --boot-disk-size "10" --boot-disk-type "pd-standard" --boot-disk-device-name "dadoonet-1"

$ gcloud compute instances create "dadoonet-2" --machine-type "n1-standard-1" \
  --scopes "https://www.googleapis.com/auth/cloud-platform" \
  --image "debian-9-stretch-v20171129" --image-project "debian-cloud" \
  --boot-disk-size "10" --boot-disk-type "pd-standard" --boot-disk-device-name "dadoonet-2"

$ gcloud compute instances create "dadoonet-3" --machine-type "n1-standard-1" \
  --scopes "https://www.googleapis.com/auth/cloud-platform" \
  --image "debian-9-stretch-v20171129" --image-project "debian-cloud" \
  --boot-disk-size "10" --boot-disk-type "pd-standard" --boot-disk-device-name "dadoonet-3"

$ gcloud compute instances list
```

NAME	ZONE	MACHINE_TYPE	PREEMPTIBLE	INTERNAL_IP	EXTERNAL_IP	STATUS
dadoonet-1	europe-west1-b	n1-standard-1		10.240.0.2	35.205.85.104	RUNNING
dadoonet-2	europe-west1-b	n1-standard-1		10.240.0.3	35.205.207.197	RUNNING
dadoonet-3	europe-west1-b	n1-standard-1		10.240.0.4	35.205.179.255	RUNNING

Manual deployment

```
# SSH
$ gcloud compute ssh dadoonet-1

# Install Java
$ sudo apt-get install default-jdk

# Install Elasticsearch
$ wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
$ sudo apt-get install apt-transport-https
$ echo "deb https://artifacts.elastic.co/packages/6.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-6.x.list
$ sudo apt-get update && sudo apt-get install elasticsearch

# Automatic startup
$ sudo /bin/systemctl daemon-reload
$ sudo /bin/systemctl enable elasticsearch.service
```

Manual deployment

```
# Change elasticsearch settings
$ sudo vi /etc/elasticsearch/elasticsearch.yml

path.data: /var/lib/elasticsearch
path.logs: /var/log/elasticsearch
node.name: dadoonet1
network.host: _site_
discovery.zen.ping.unicast.hosts: ["10.240.0.2", "10.240.0.3", "10.240.0.4"]
discovery.zen.minimum_master_nodes: 2

# Start elasticsearch
sudo systemctl start elasticsearch.service

# Check logs
sudo tail -f /var/log/elasticsearch/elasticsearch.log

[2017-12-11T16:14:35,707][INFO ][o.e.n.Node                               ] [dadoonet1] started

# Try elasticsearch
curl '10.240.0.2:9200/?pretty'
```


Check cluster nodes

```
$ curl 10.240.0.2:9200/_cat/nodes?v
```

ip	heap.percent	ram.percent	cpu	load_1m	load_5m	load_15m	node.role	master	name
10.240.0.2	8	64	5	0.03	0.05	0.05	mdi	*	dadoonet1
10.240.0.3	6	63	13	0.23	0.10	0.04	mdi	-	dadoonet2
10.240.0.4	6	64	4	0.07	0.11	0.08	mdi	-	dadoonet3



elasticsearch

Use discovery-gce plugin

GCE Discovery setup

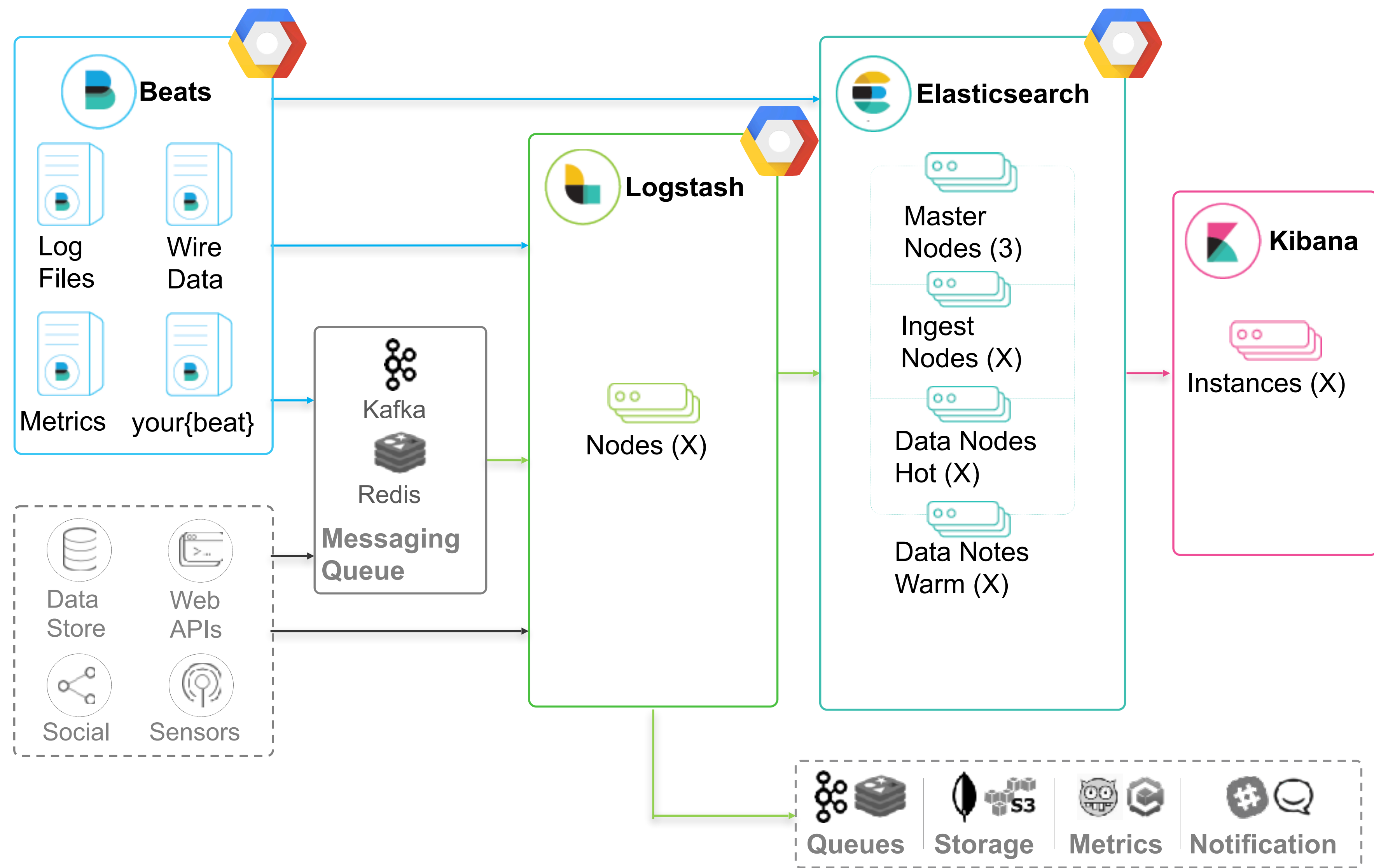
```
# Install GCE Discovery plugin
$ sudo /usr/share/elasticsearch/bin/elasticsearch-plugin install discovery-gce

# Change elasticsearch settings
$ sudo vi /etc/elasticsearch/elasticsearch.yml
# Remove the following line
# discovery.zen.ping.unicast.hosts: ["10.240.0.2", "10.240.0.3", "10.240.0.4"]
# And add
cloud.gce.project_id: dadoonet-1082
cloud.gce.zone: europe-west1-b
discovery.zen.hosts_provider: gce

# Restart elasticsearch
sudo systemctl stop elasticsearch.service
sudo systemctl start elasticsearch.service

# Check logs
sudo tail -f /var/log/elasticsearch/elasticsearch.log
```

Elastic Platform





kibana

Kibana Manual deployment

Manual deployment

```
# Create a firewall rule
$ gcloud compute firewall-rules create kibana --direction=INGRESS --priority=1000 \
  --network=default --action=ALLOW --rules=tcp:5601 --source-ranges=0.0.0.0/0 \
  --target-tags=kibana

# Create Kibana GCE Instance
$ gcloud compute instances create "dadoonet-k" --machine-type "n1-standard-1" \
  --scopes "https://www.googleapis.com/auth/cloud-platform" --tags "kibana" \
  --image "debian-9-stretch-v20171129" --image-project "debian-cloud" \
  --boot-disk-size "10" --boot-disk-type "pd-standard" --boot-disk-device-name "dadoonet-k"

$ gcloud compute instances list
```

NAME	ZONE	MACHINE_TYPE	PREEMPTIBLE	INTERNAL_IP	EXTERNAL_IP	STATUS
dadoonet-1	europe-west1-b	n1-standard-1		10.240.0.2	35.205.85.104	RUNNING
dadoonet-2	europe-west1-b	n1-standard-1		10.240.0.3	35.205.207.197	RUNNING
dadoonet-3	europe-west1-b	n1-standard-1		10.240.0.4	35.205.179.255	RUNNING
dadoonet-k	europe-west1-b	n1-standard-1		10.240.0.5	35.195.167.220	RUNNING

Manual deployment

```
# SSH
$ gcloud compute ssh dadoonet-k

# Install Elasticsearch
$ wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
$ sudo apt-get install apt-transport-https
$ echo "deb https://artifacts.elastic.co/packages/6.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-6.x.list
$ sudo apt-get update && sudo apt-get install kibana

# Automatic startup
$ sudo /bin/systemctl daemon-reload
$ sudo /bin/systemctl enable kibana.service
```


Manual deployment

```
# Change kibana settings
$ sudo vi /etc/kibana/kibana.yml

server.host: "10.240.0.5"
elasticsearch.url: "http://10.240.0.2:9200"

# Start elasticsearch
sudo systemctl start kibana.service

# Check logs
sudo journalctl -f
```



Dev Tools

History Settings Help

Console

 Discover

 Visualize

 **Dashboard**
Timelion

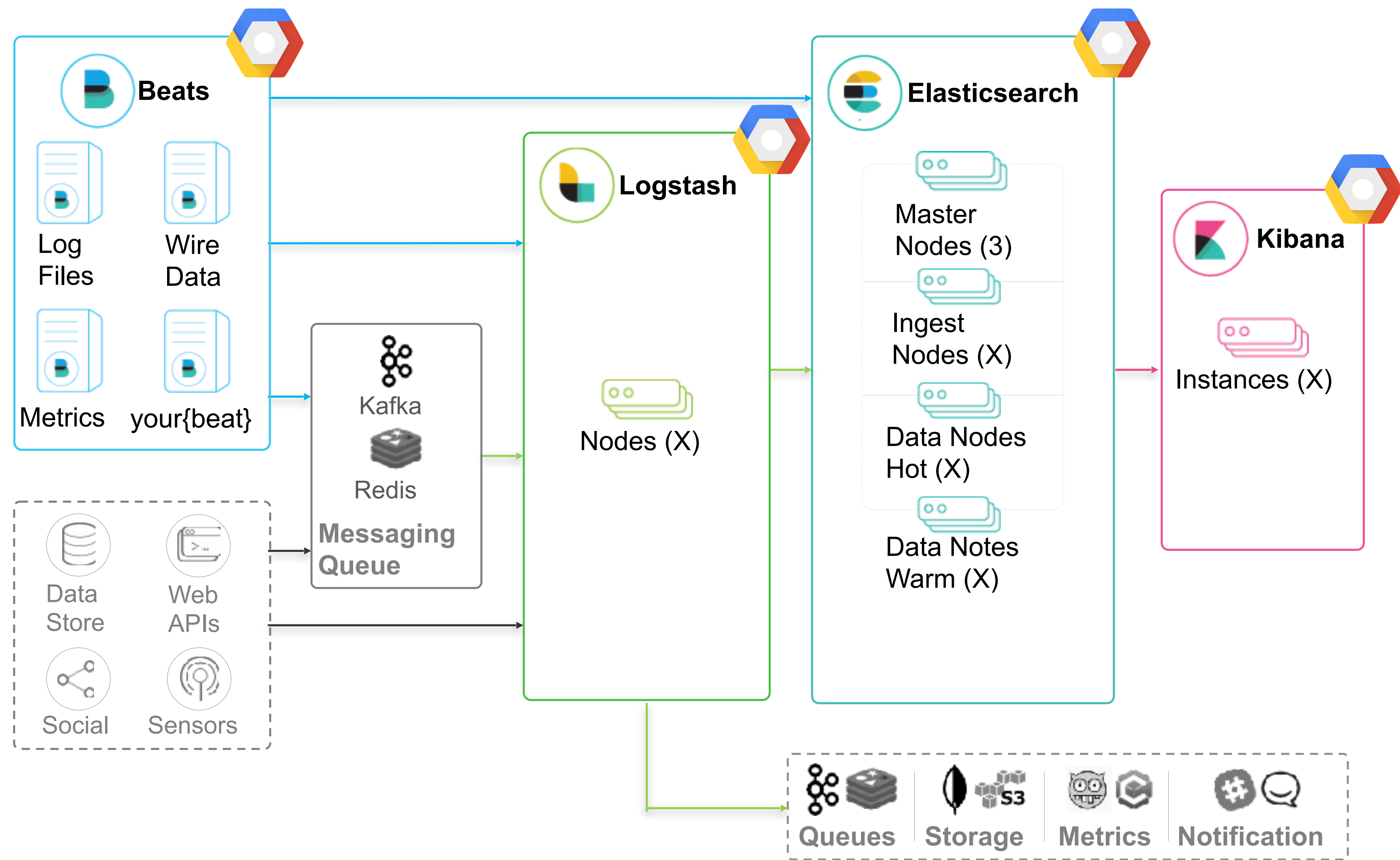
 Dev Tools

 Management

```
1 GET /
```

```
1 {
2   "name": "dadoonet1",
3   "cluster_name": "elasticsearch",
4   "cluster_uuid": "GKbNo18iTzqZcnrhbD70UQ",
5   "version": {
6     "number": "6.0.1",
7     "build_hash": "601be4a",
8     "build_date": "2017-12-04T09:29:09.525Z",
9     "build_snapshot": false,
10    "lucene_version": "7.0.1",
11    "minimum_wire_compatibility_version": "5.6.0",
12    "minimum_index_compatibility_version": "5.0.0"
13  },
14   "tagline": "You Know, for Search"
15 }
```

Elastic Platform

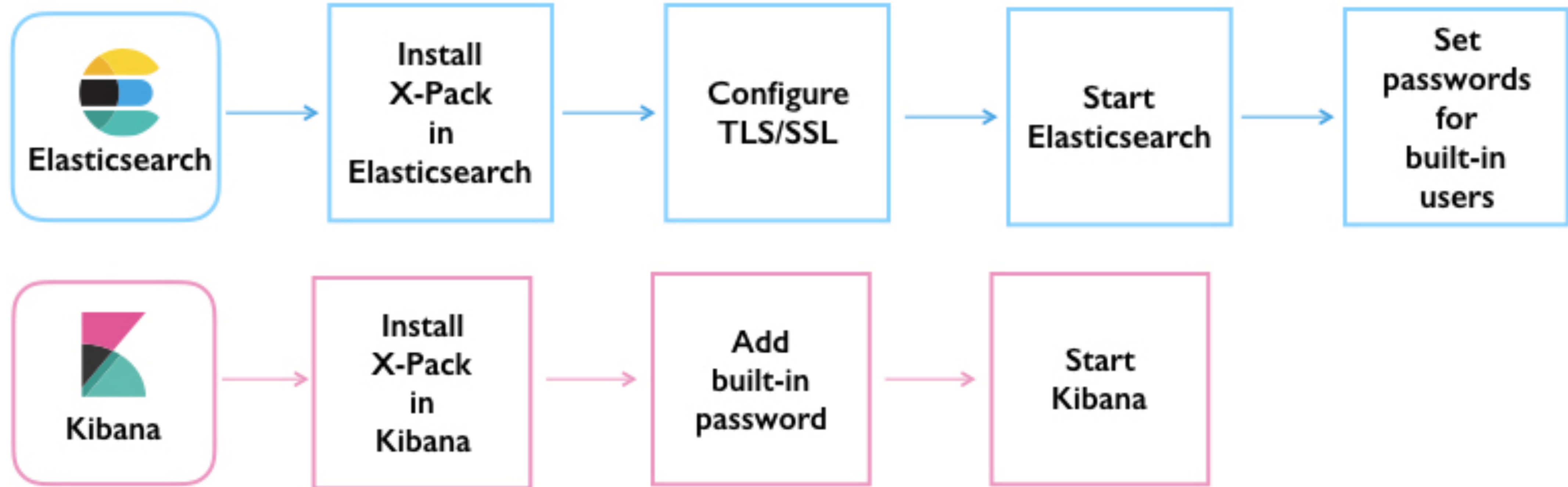




x-pack

X-Pack Manual deployment

TODO List





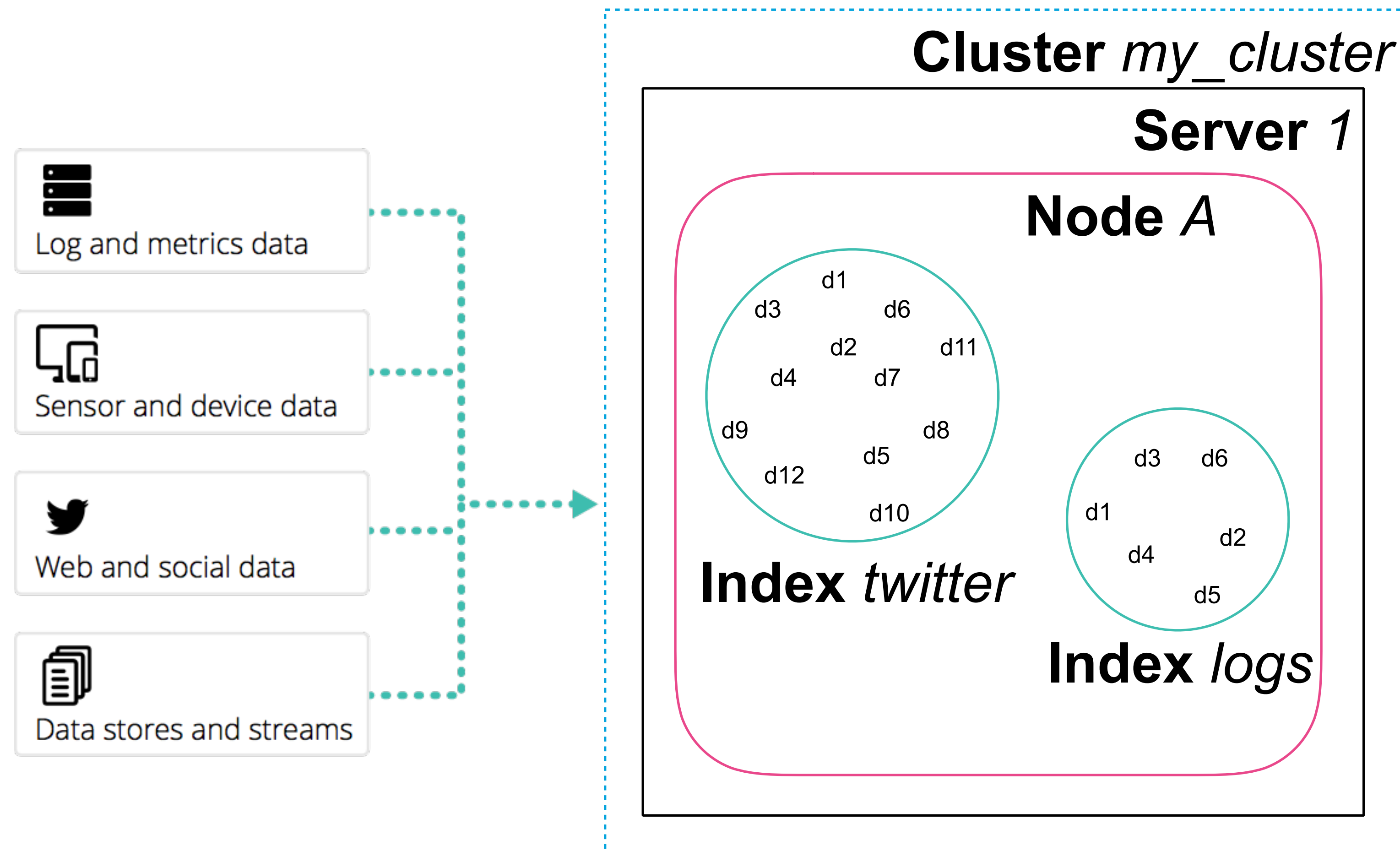
cloud

One click install
<https://cloud.elastic.co>

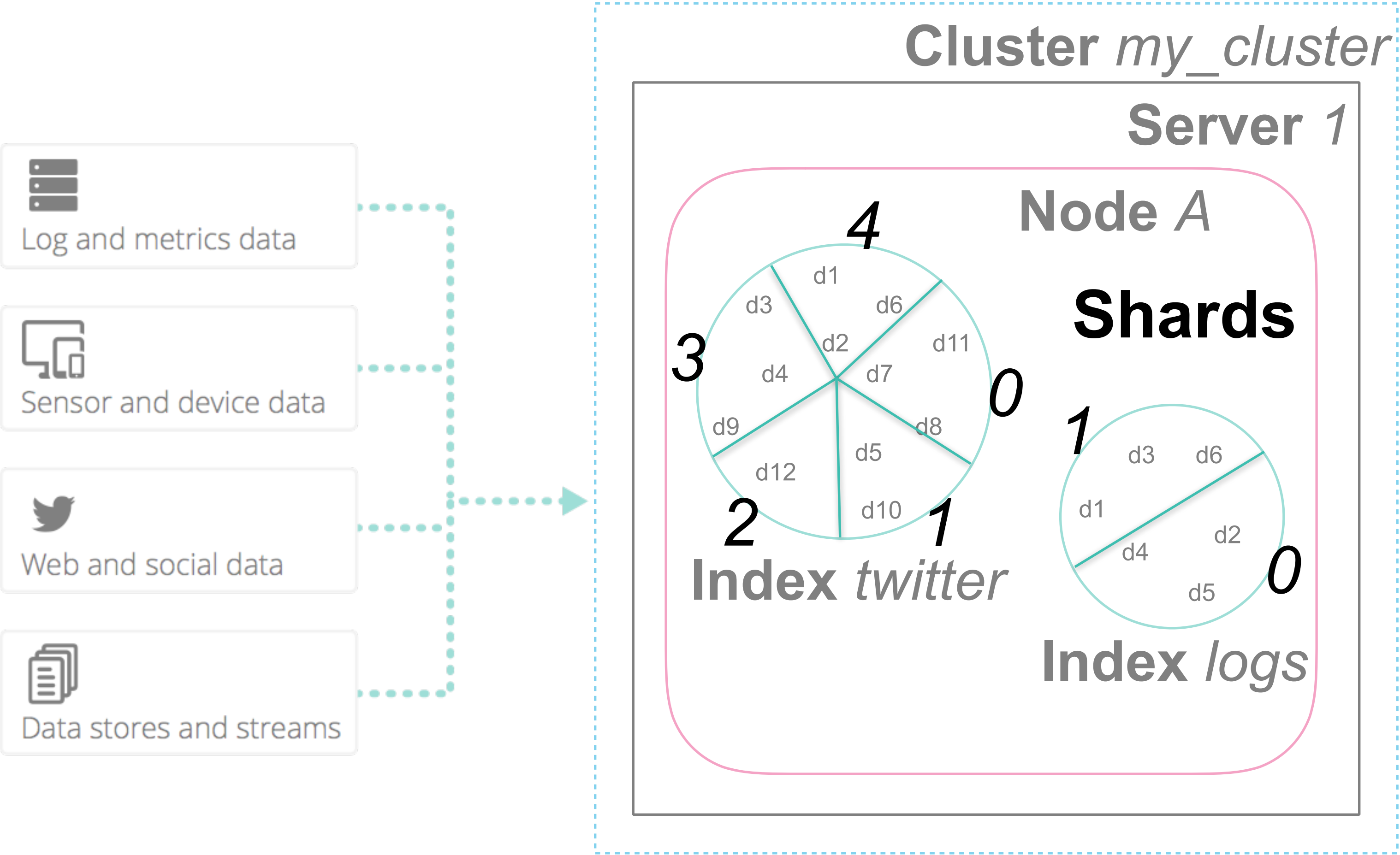


Elasticsearch Cluster Sizing

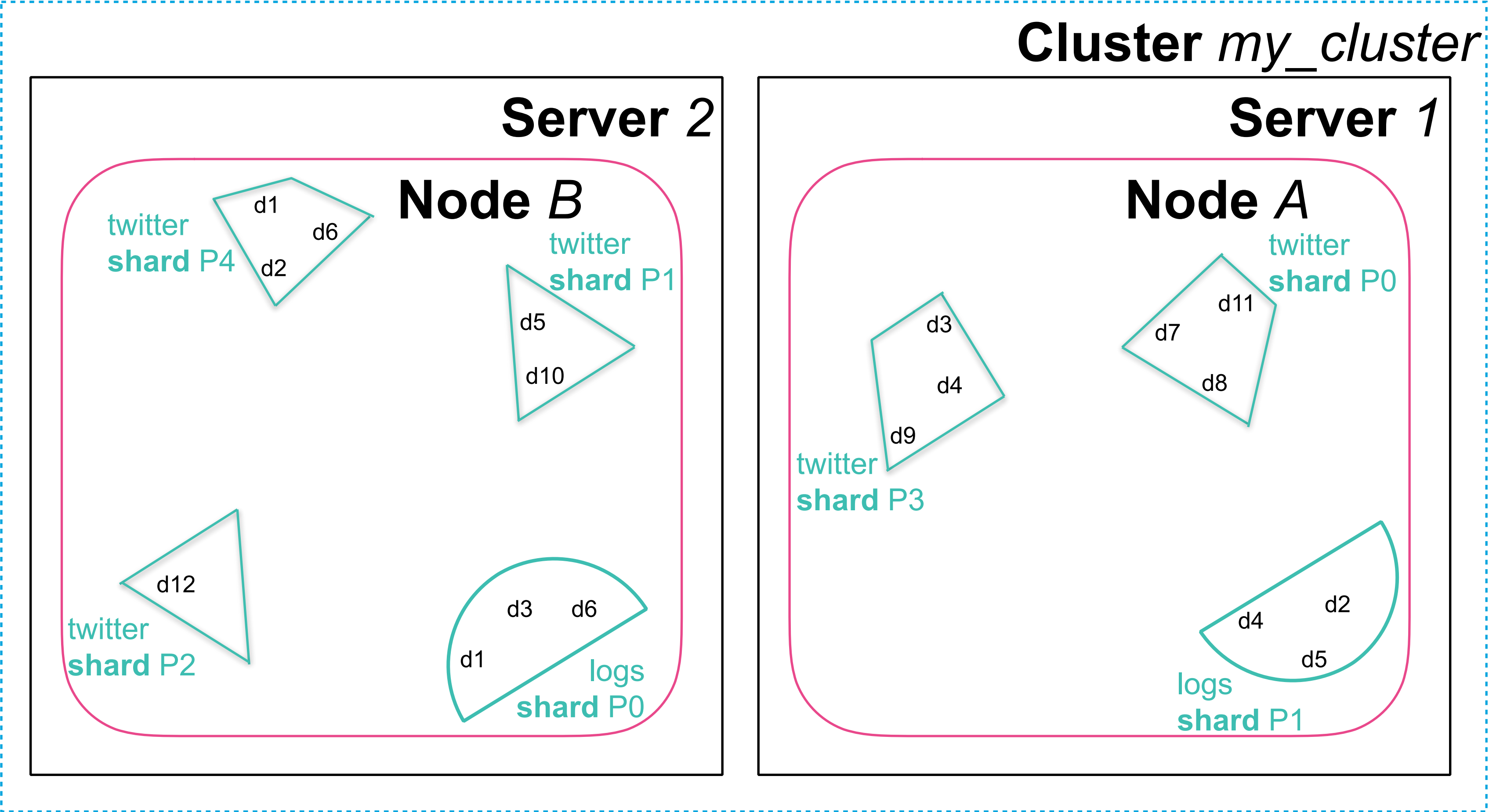
Terminology



Partition

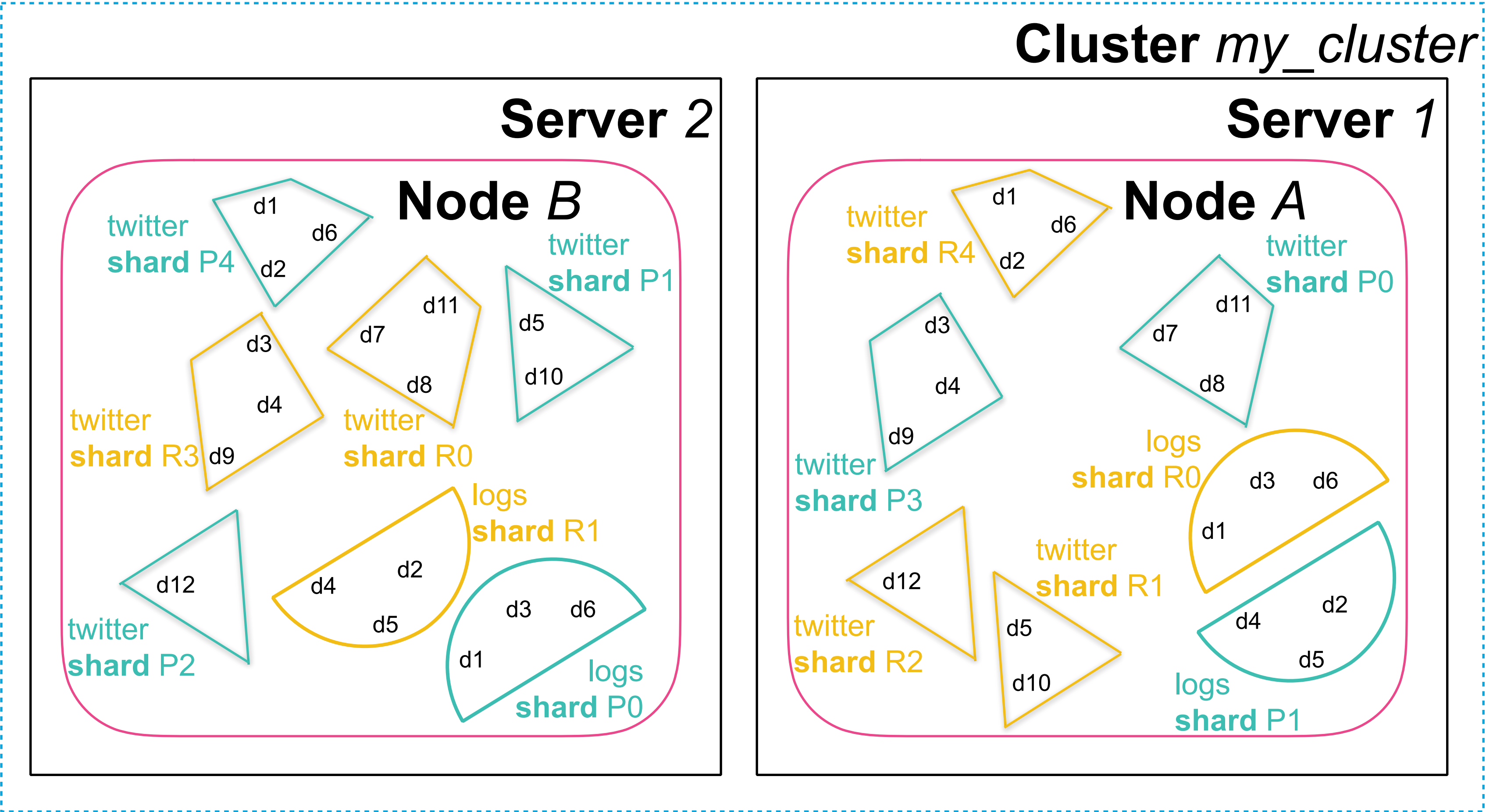


Distribution



Replication

- Primaries
- Replicas



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 $\Rightarrow \sim 7TB$

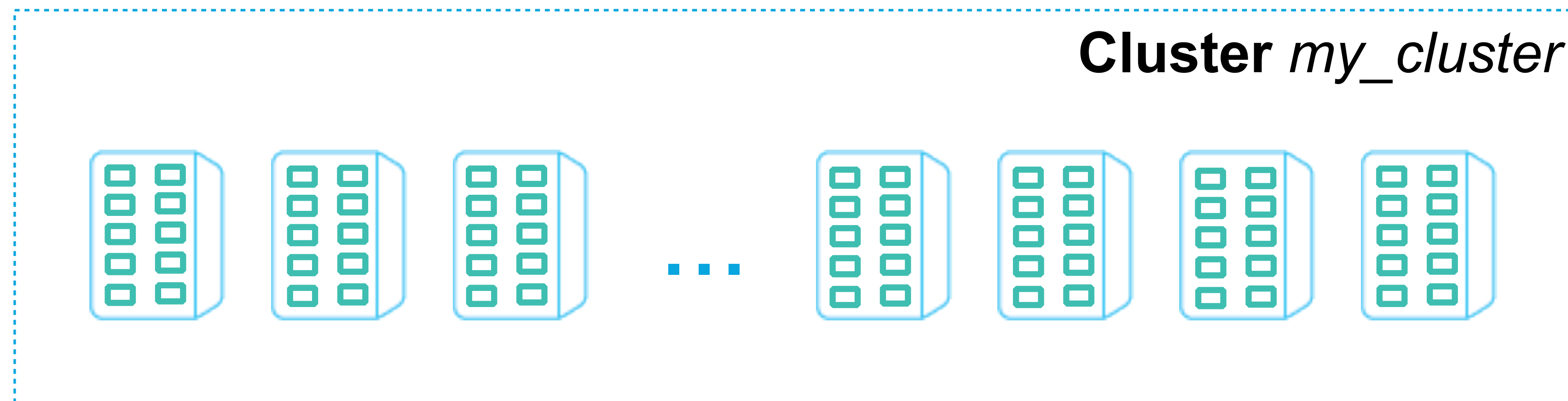
Shard Size

- 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?

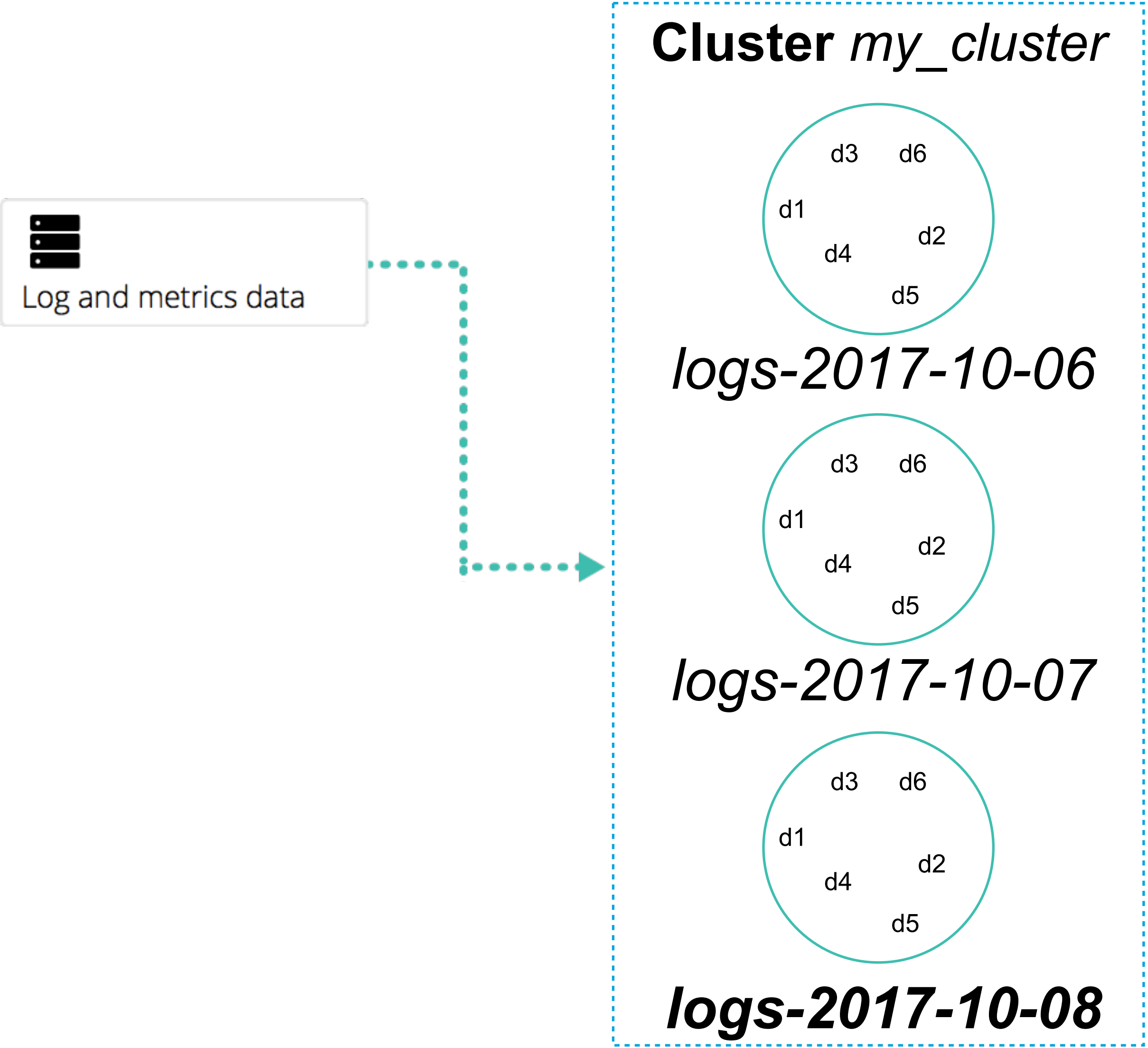
Time-Based Data

- 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



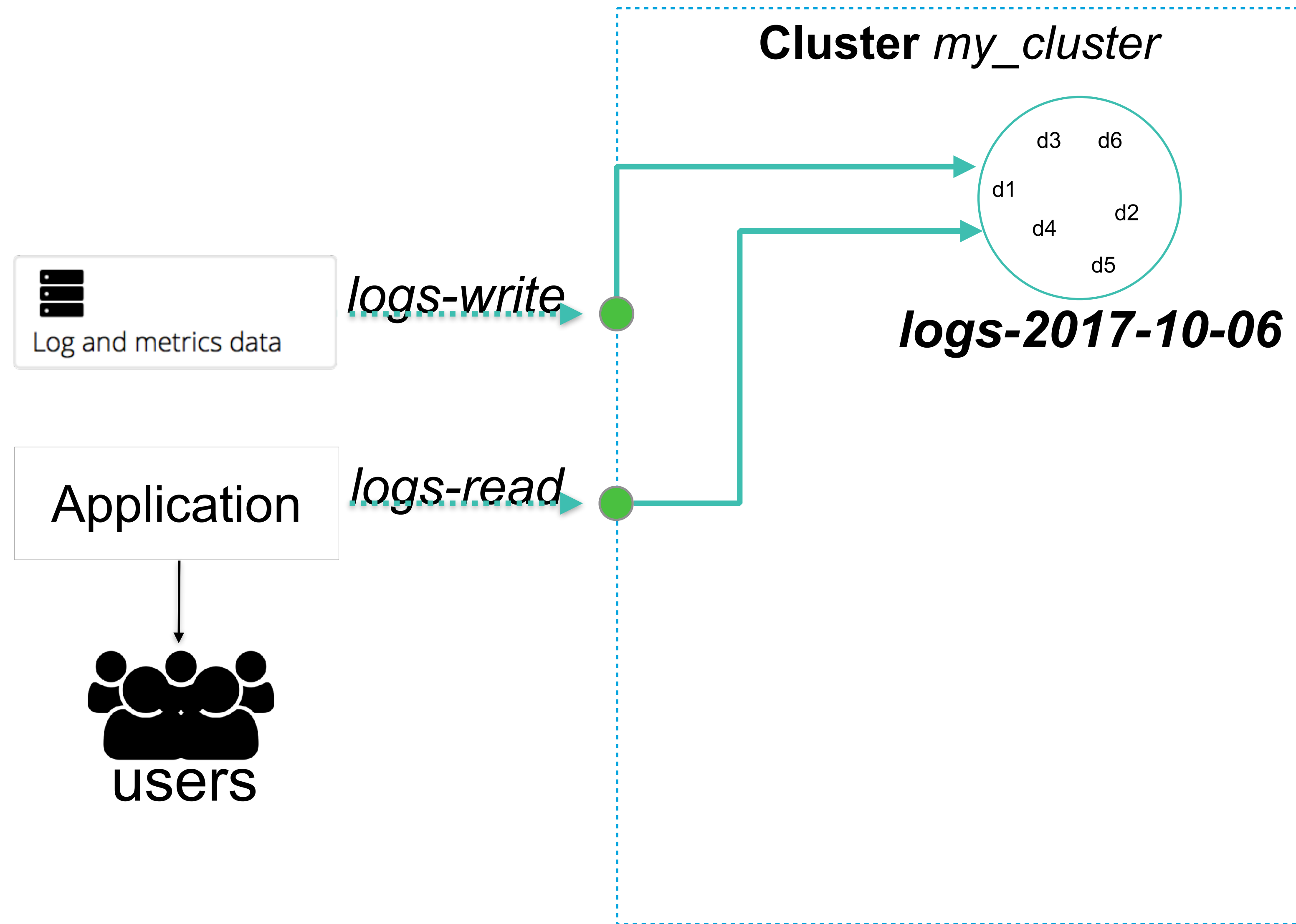
Templates

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

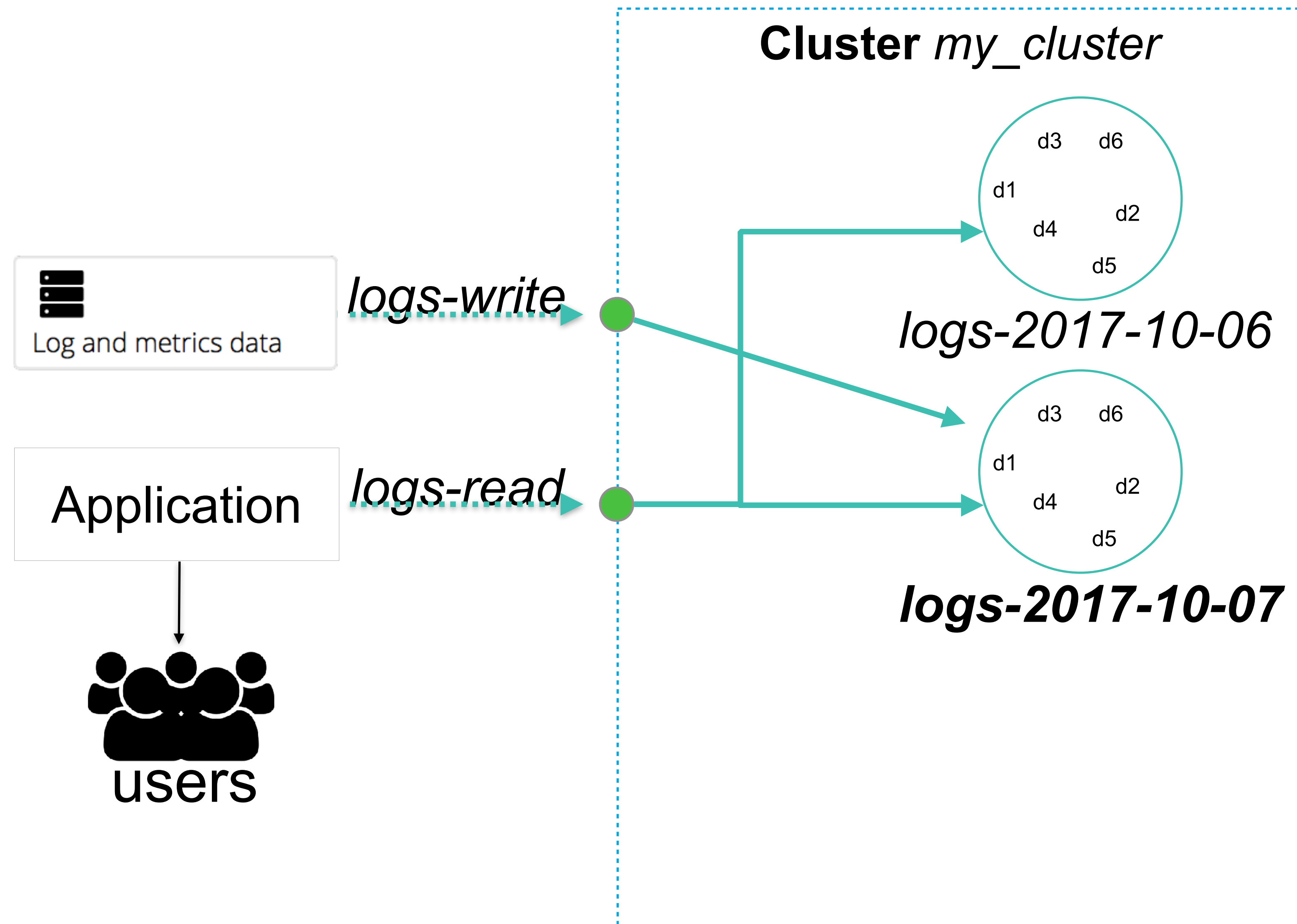
```
PUT _template/logs
{
  "template": "logs-*",
  "settings": {
    "number_of_shards": 2,
    "number_of_replicas": 1,
    "refresh_interval": "60s"
  }
}
```

More on that later

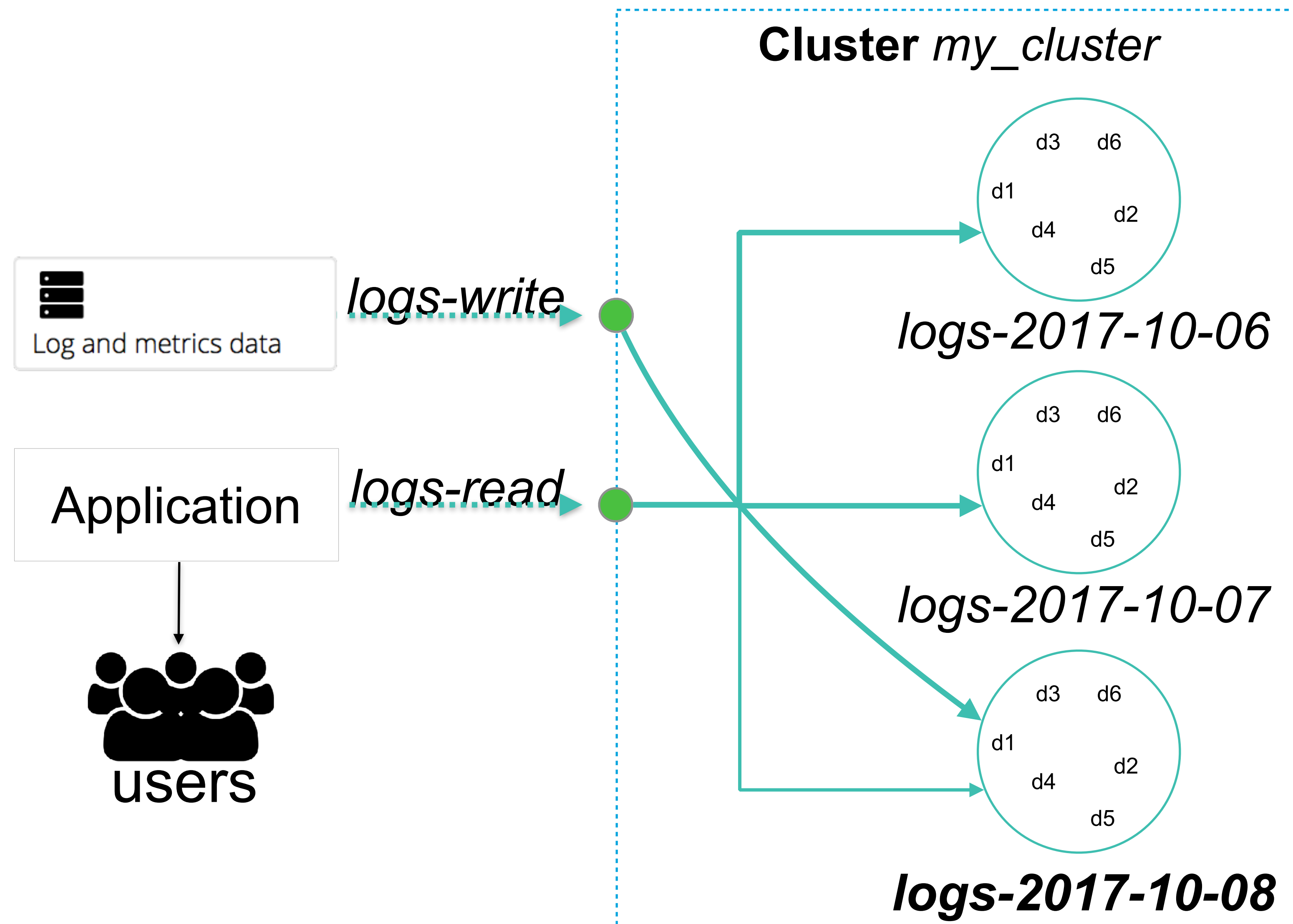
Alias



Alias



Alias





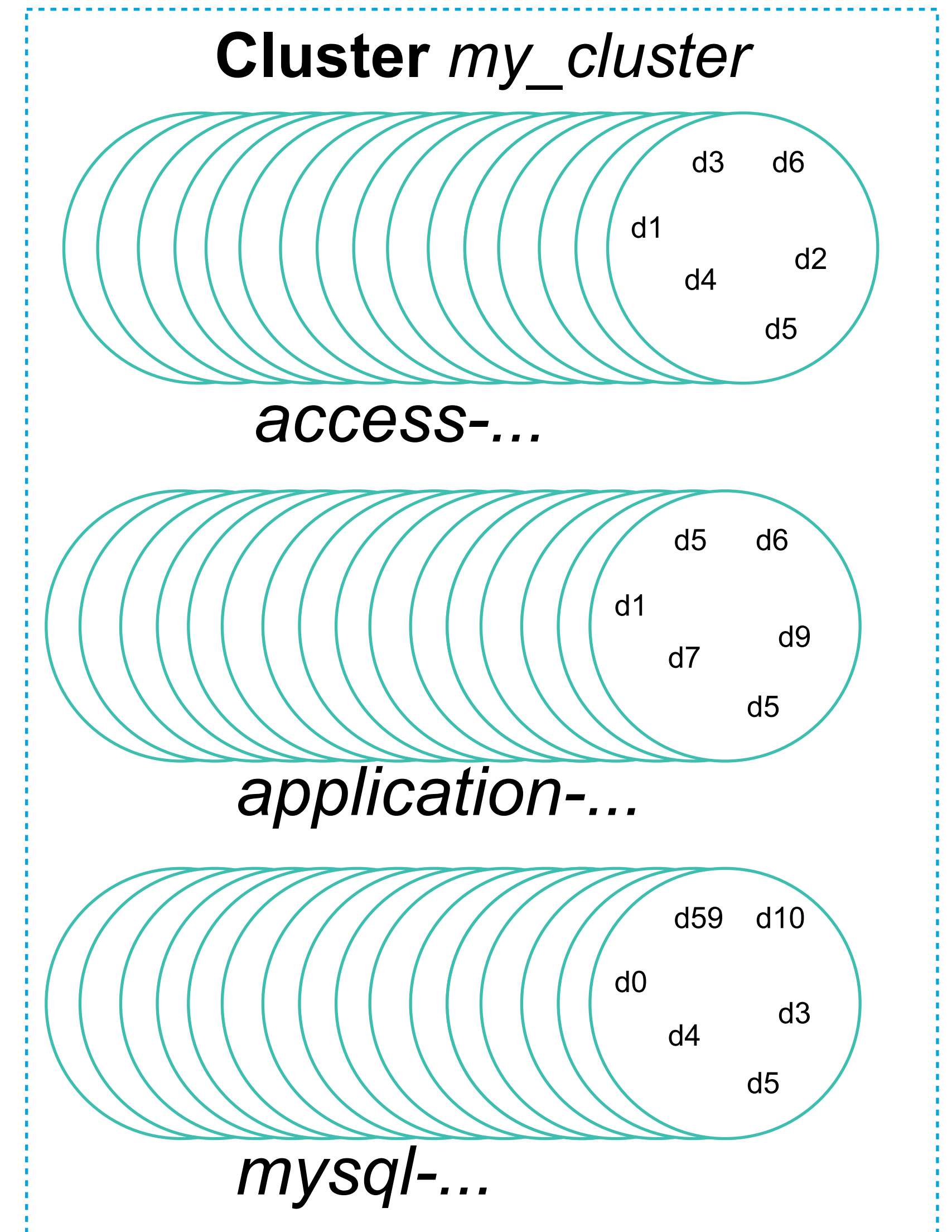
Detour: Rollover API

<https://www.elastic.co/guide/en/elasticsearch/reference/5.6/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
```




Detour: Shrink API

<https://www.elastic.co/guide/en/elasticsearch/reference/5.6/indices-shrink-index.html>

Scaling



But what happens if we have 2M users?

Scaling



Scaling



Scaling



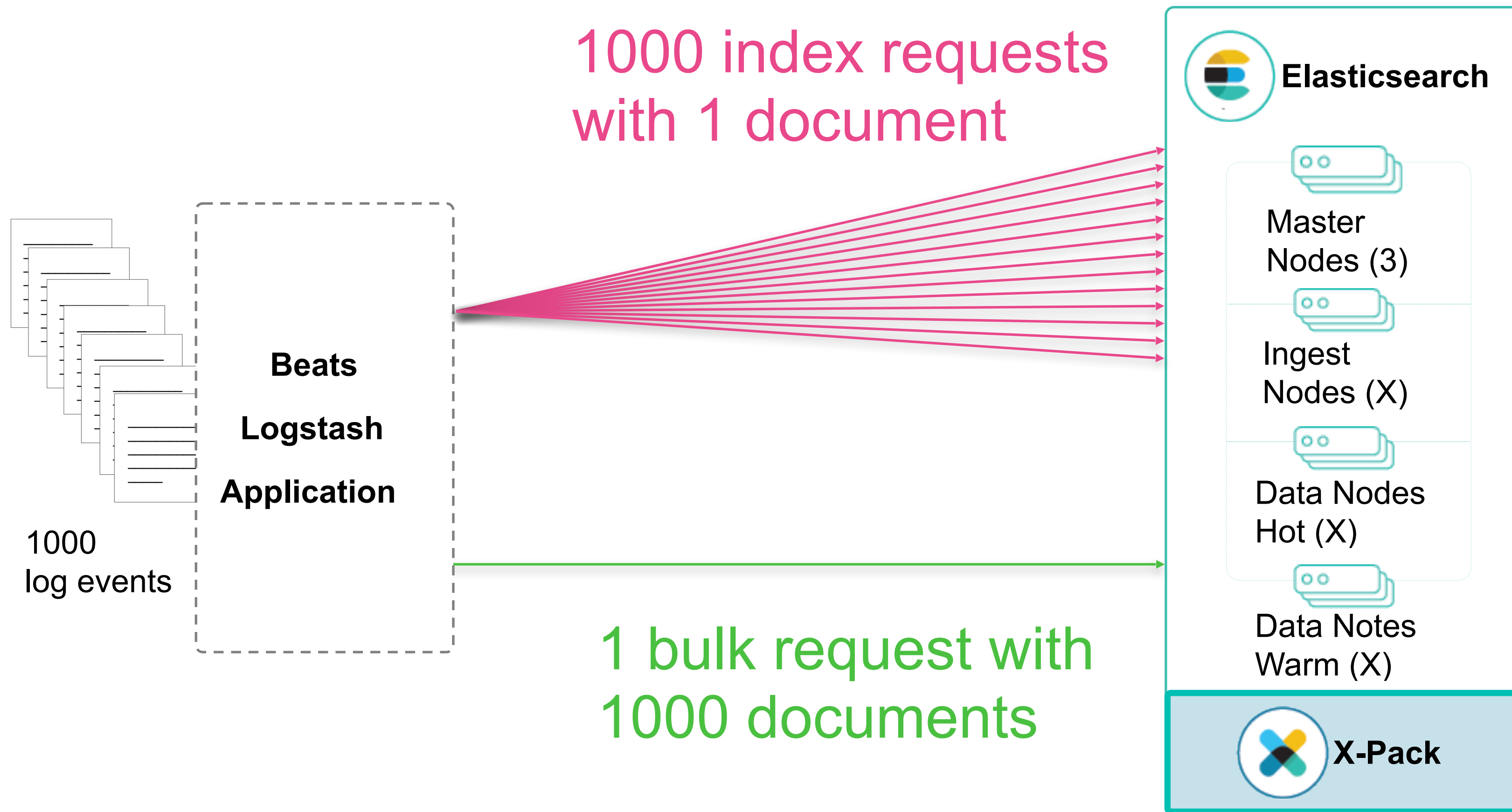
Shards are the working unit

- 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**)

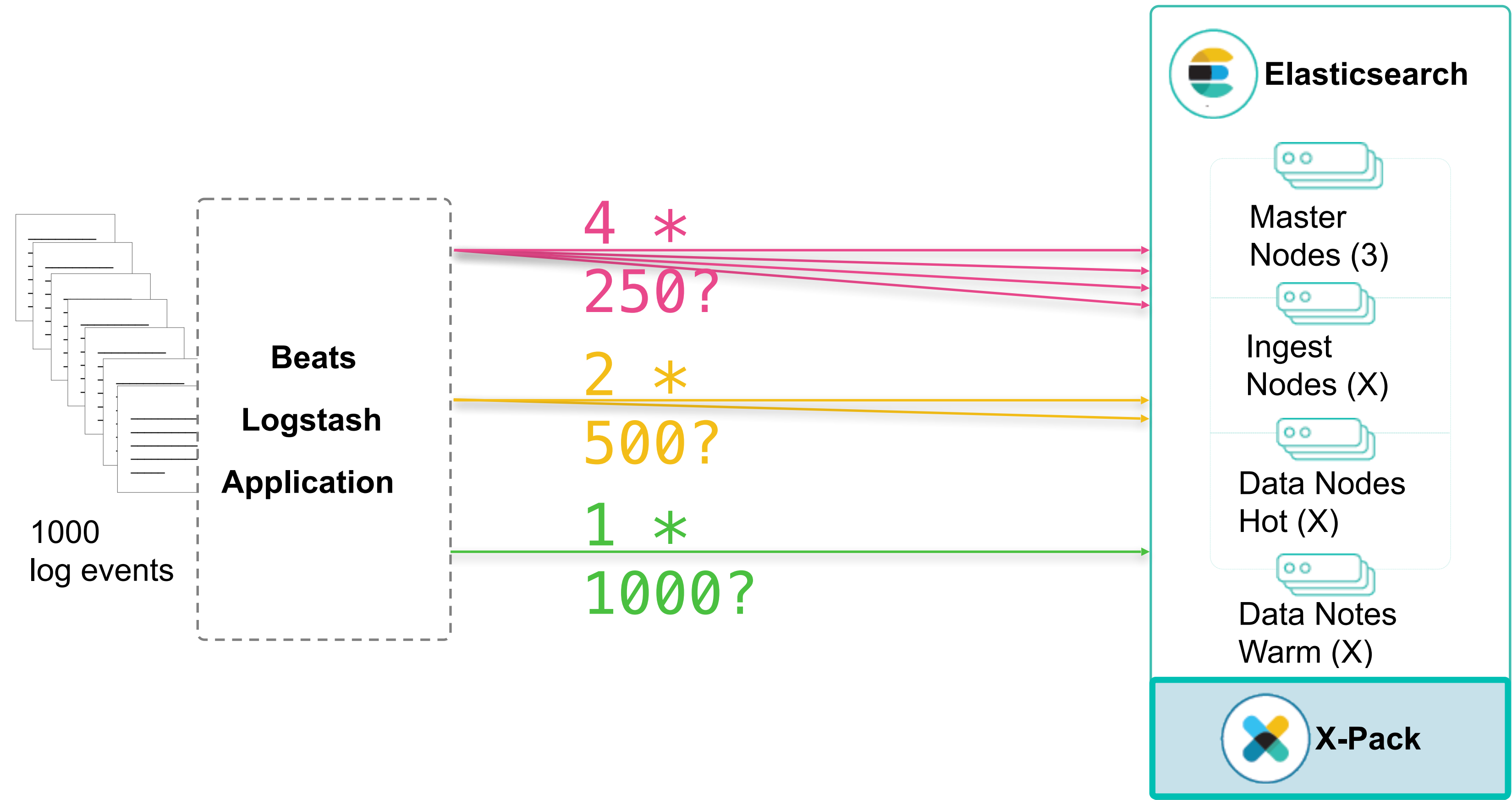


Optimal Bulk Size

What is Bulk?



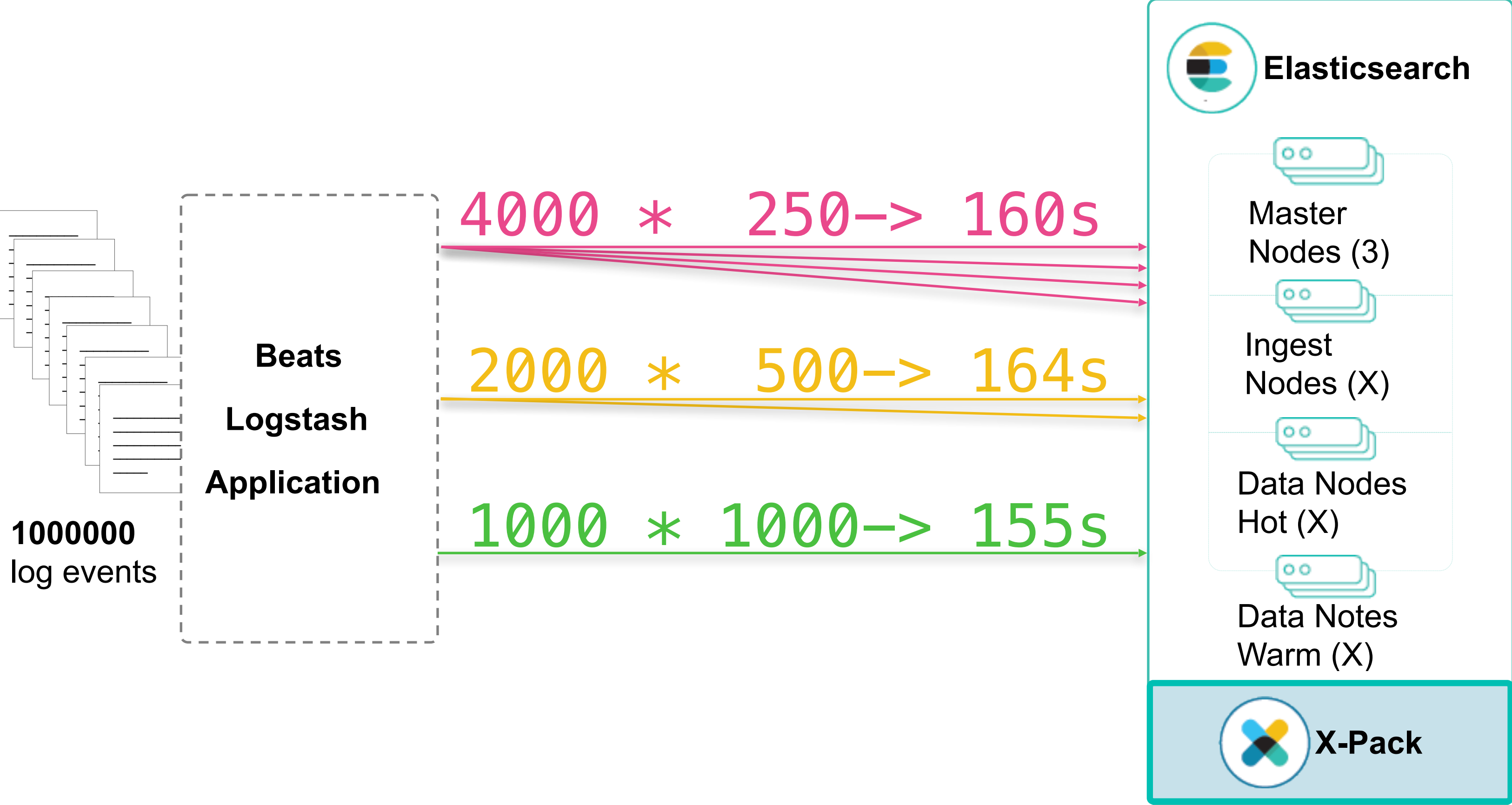
What is the optimal bulk size?



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
}

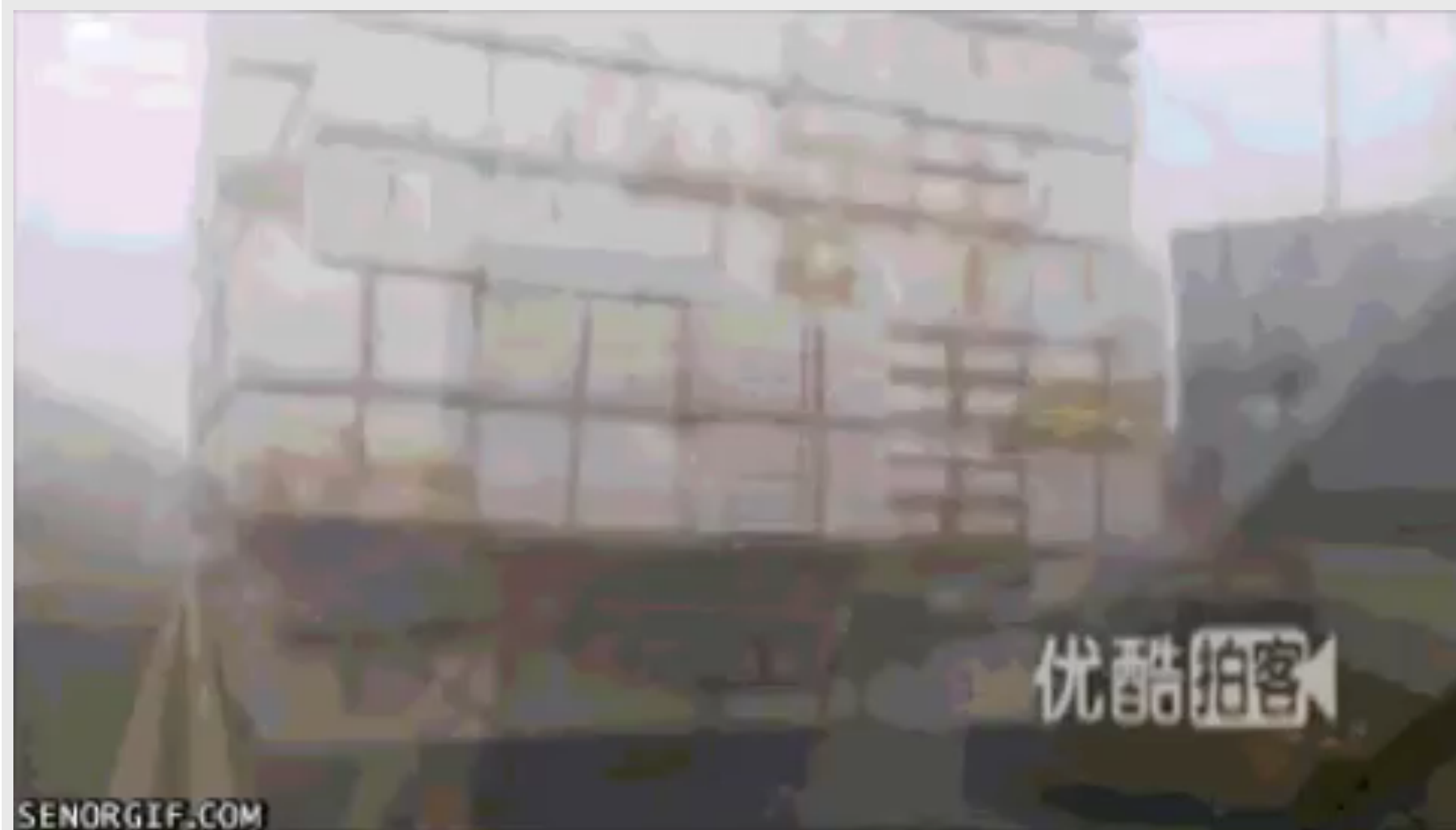
for SIZE in 100 500 1000 3000 5000 10000; do
  for i in {1..20}; do
    exec_test $SIZE
  done; done;
```


Test it ;)

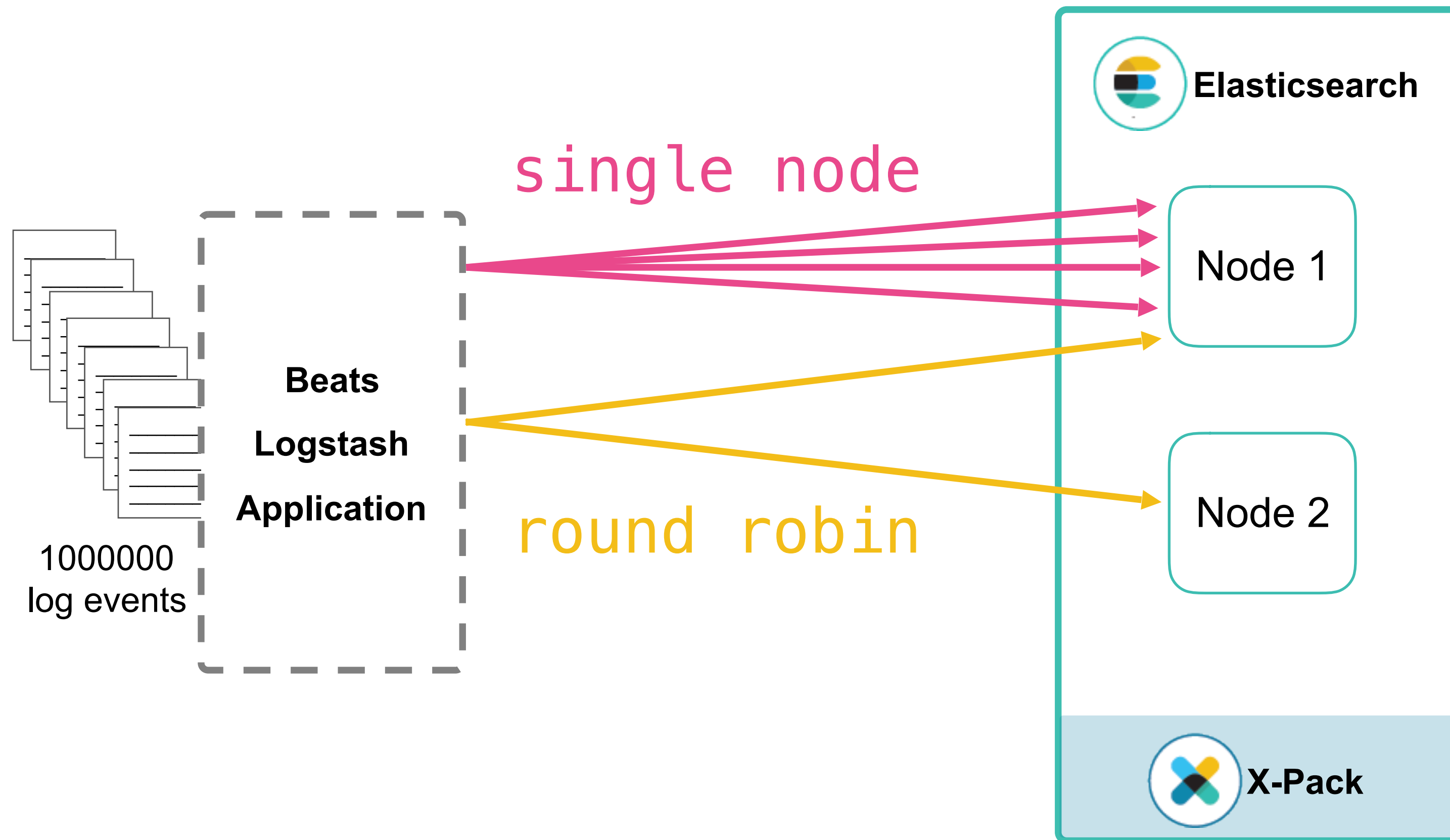
- 2 node cluster (m3.large)
 - 2 vCPU, 7.5GB Memory, 1x32GB SSD
- 1 index server (m3.large)
 - logstash
 - kibana

# 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

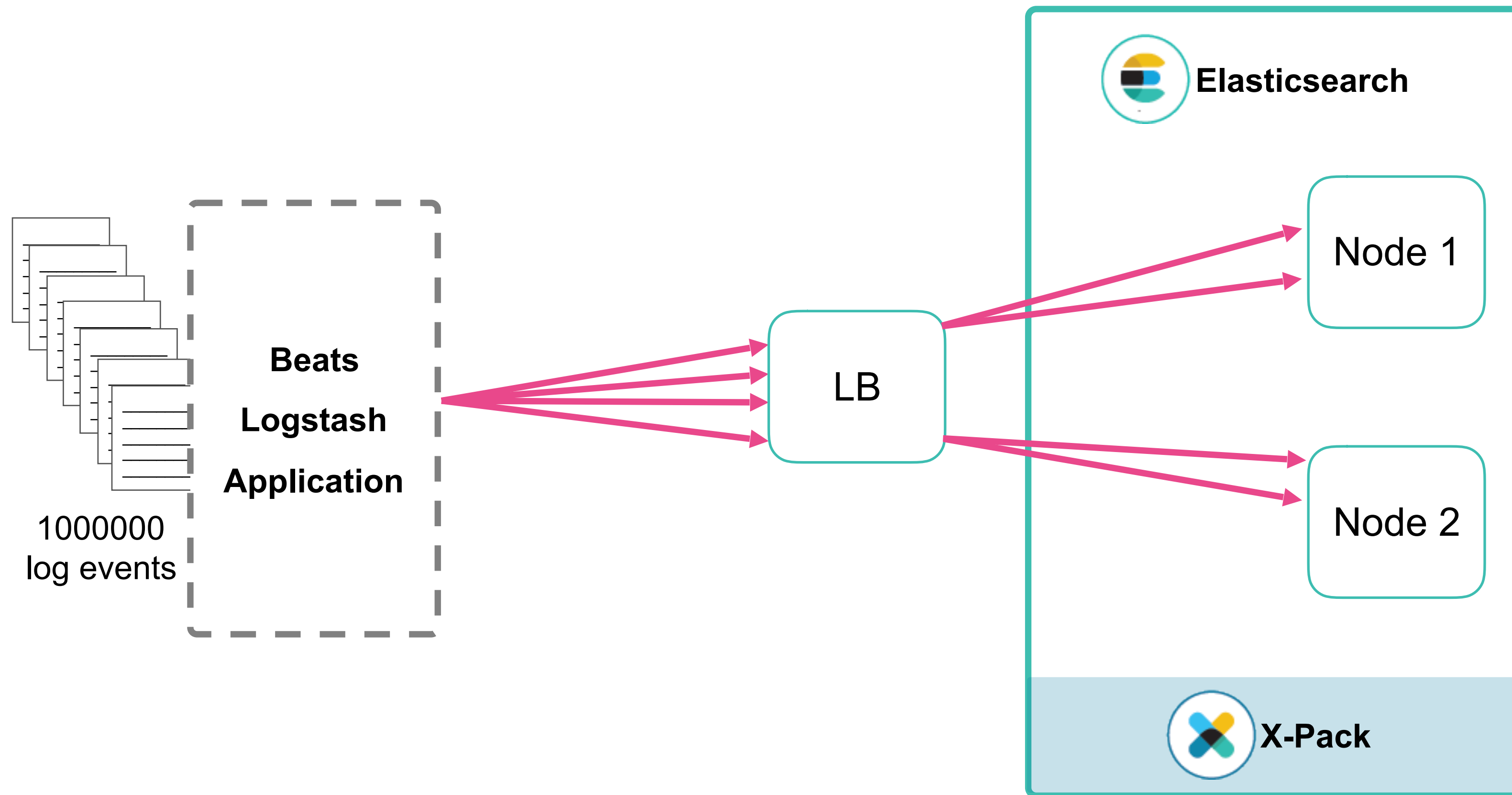


Clients

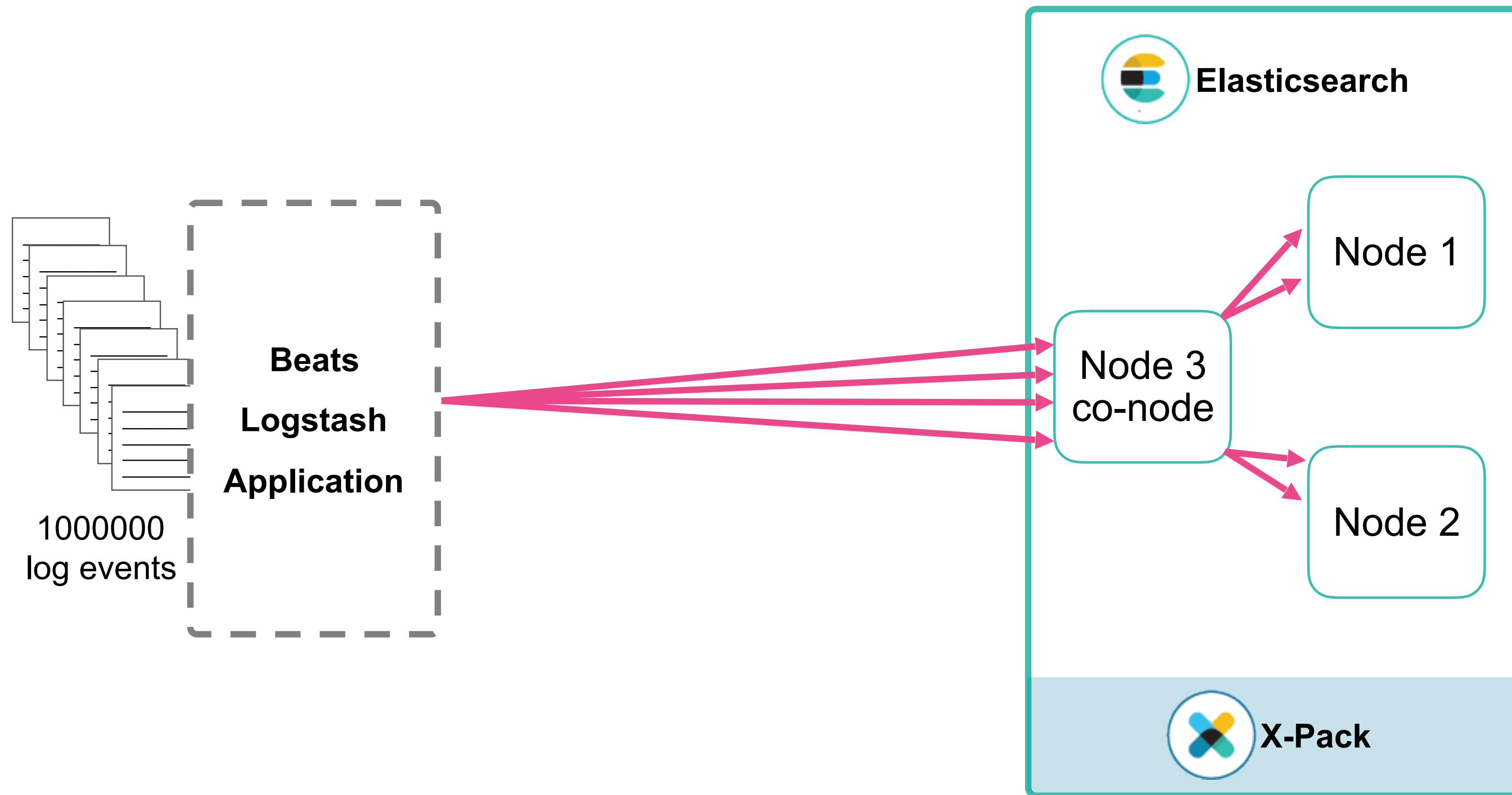
- Most clients implement round robin
 - you specify a seed list
 - the client sniffs the cluster
 - the client implement different selectors
- Logstash allows an array (no sniffing)
- Beats allows an array (no sniffing)
- Kibana only connects to one single node

```
output {  
  elasticsearch {  
    hosts => ["node1","node2","node3"]  
  }  
}
```


Load Balancer



Coordinating-only Node



Test it ;)

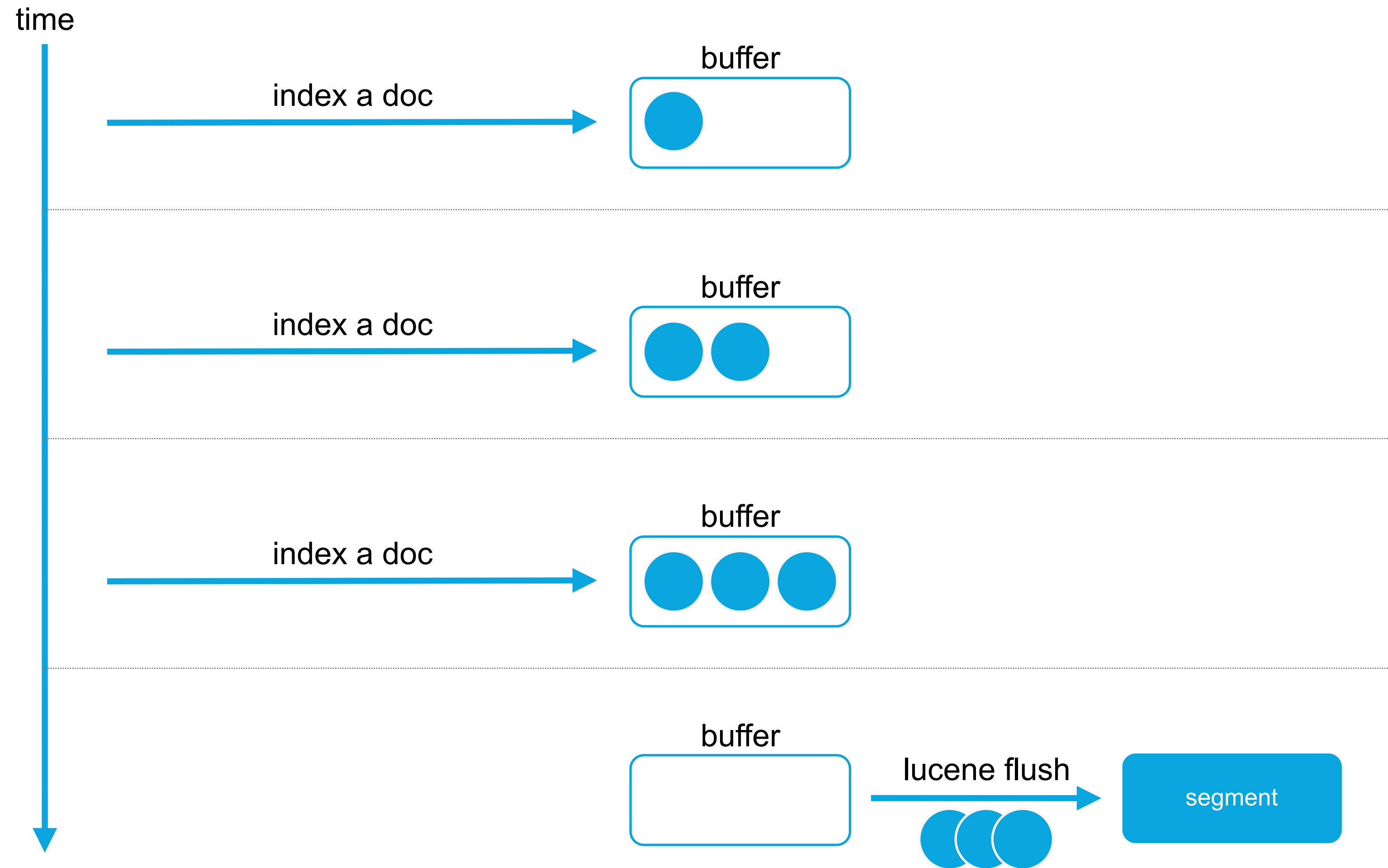
- 2 node cluster (m3.large)
 - 2 vCPU, 7.5GB Memory, 1x32GB SSD
- 1 index server (m3.large)
 - logstash (round robin configured)
 - `hosts => ["10.12.145.189", "10.121.140.167"]`
 - kibana

#docs time(s)	100	500	1000
NO Round Robin	191.7	161.9	163.5
Round Robin	189.7	159.7	159.0



Optimizing Disk IO

Durability



refresh_interval

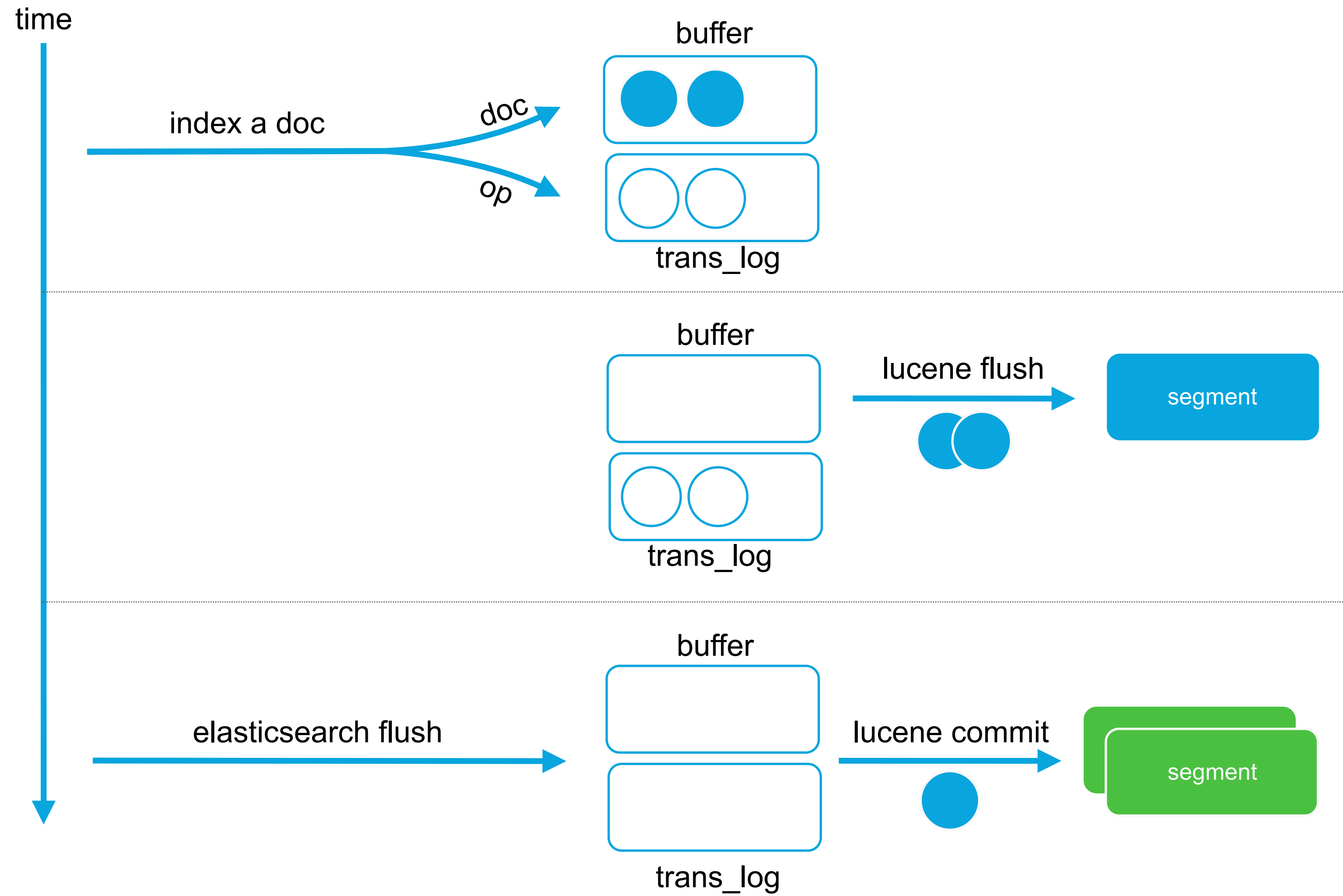
- Dynamic per-index setting

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

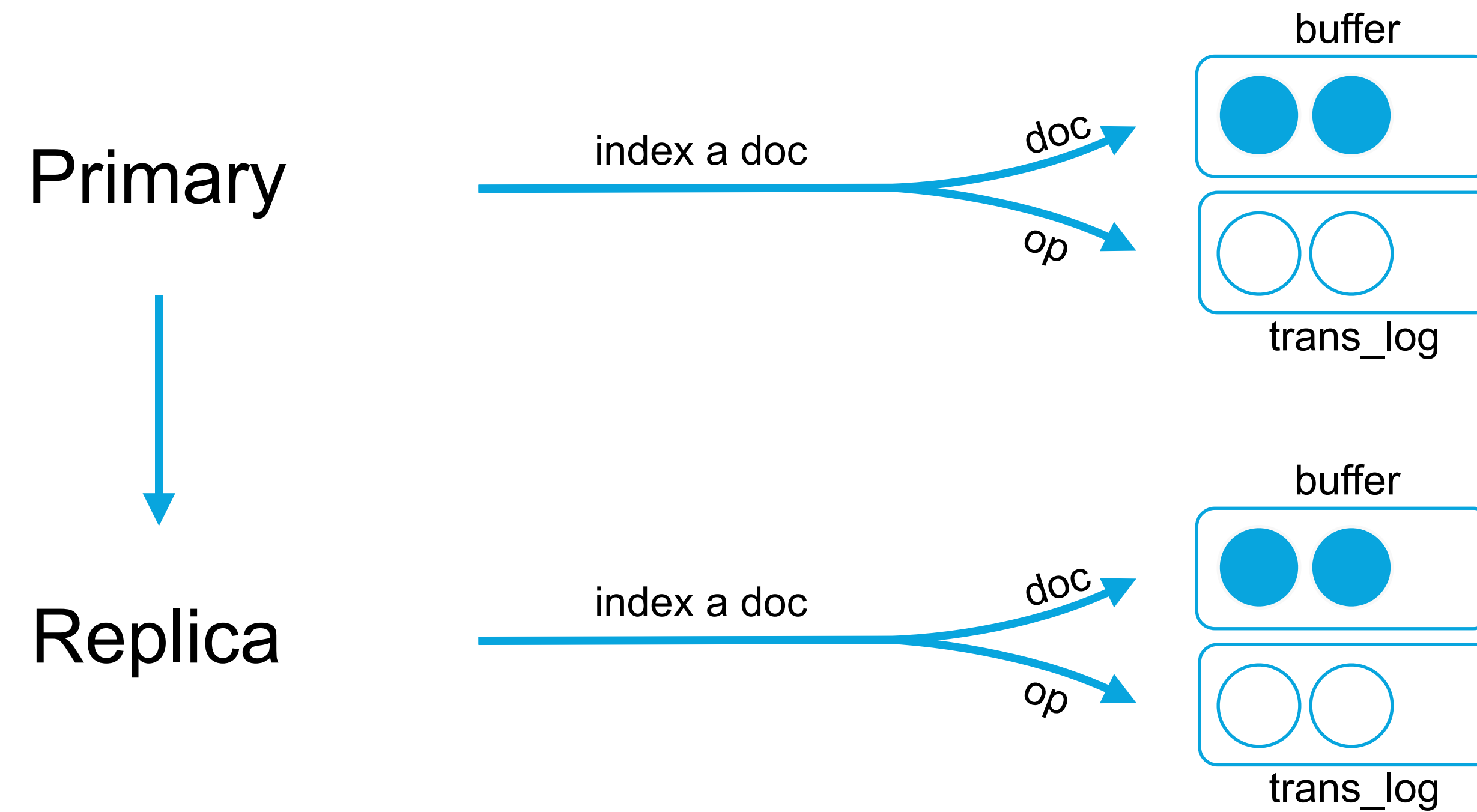
- Increase to get better write throughput to an index
- New documents will take more time to be available for Search.

#docs time(s)	100	500	1000
1s refresh	189.7	159.7	159.0
60s refresh	185.8	152.1	152.6

Durability



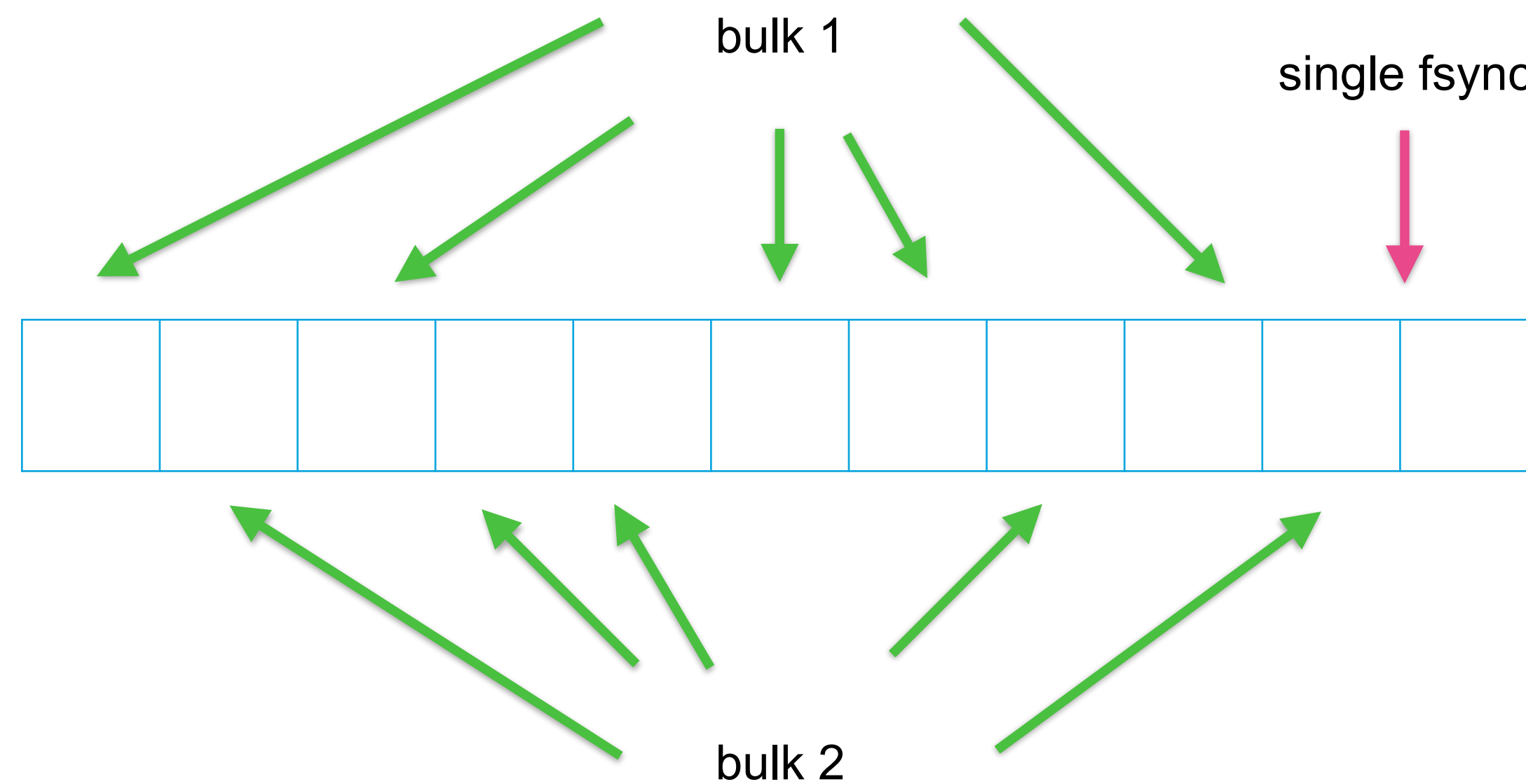
Translog fsync every 5s (1.7)



redundancy doesn't help if all nodes lose power

Translog fsync on every request

- For low volume indexing, fsync matters less
- For high volume indexing, we can amortize the costs and fsync on every bulk
- Concurrent requests can share an fsync



Async Transaction Log

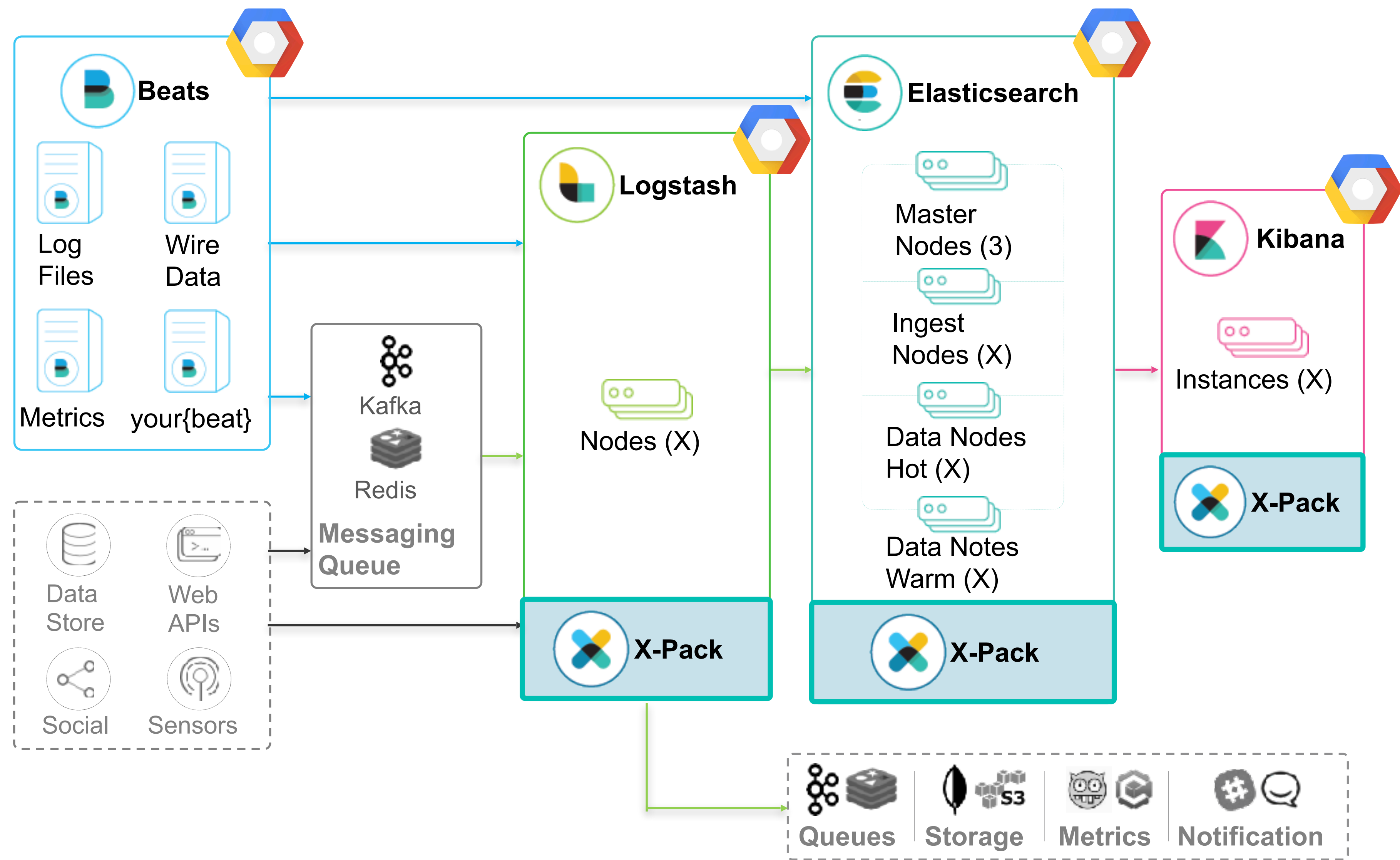
- `index.translog.durability`
 - request (default)
 - async
- `index.translog.sync_interval` (only if async is set)
- Dynamic per-index settings
- **Be careful, you are relaxing the safety guarantees**

#docs time(s)	100	500	1000
Request fsync	185.8	152.1	152.6
5s sync	154.8	143.2	143.1



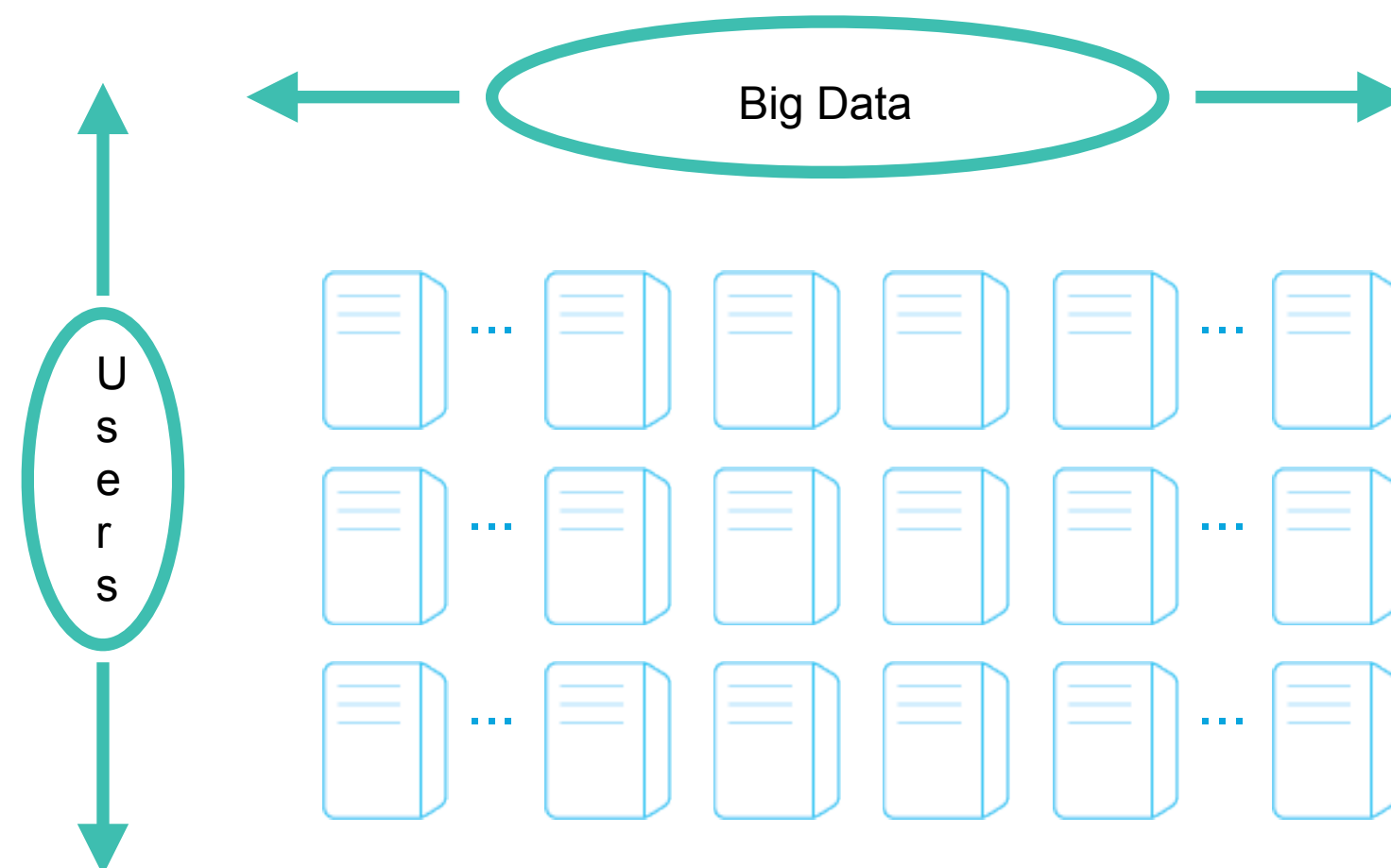
Final Remarks

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	100	500	1000
Default	191.7s	161.9s	163.5s
RR+60s+Async5s	154.8s	143.2s	143.1s



Déployer et monitorer avec elastic sur GCP

David Pilato
Developer | Evangelist, @dadoonet

