

Module

Observability Fundamentals



Topics

- Observability with the Elastic Stack
- Logs
- Metrics
- APM

Lesson 1

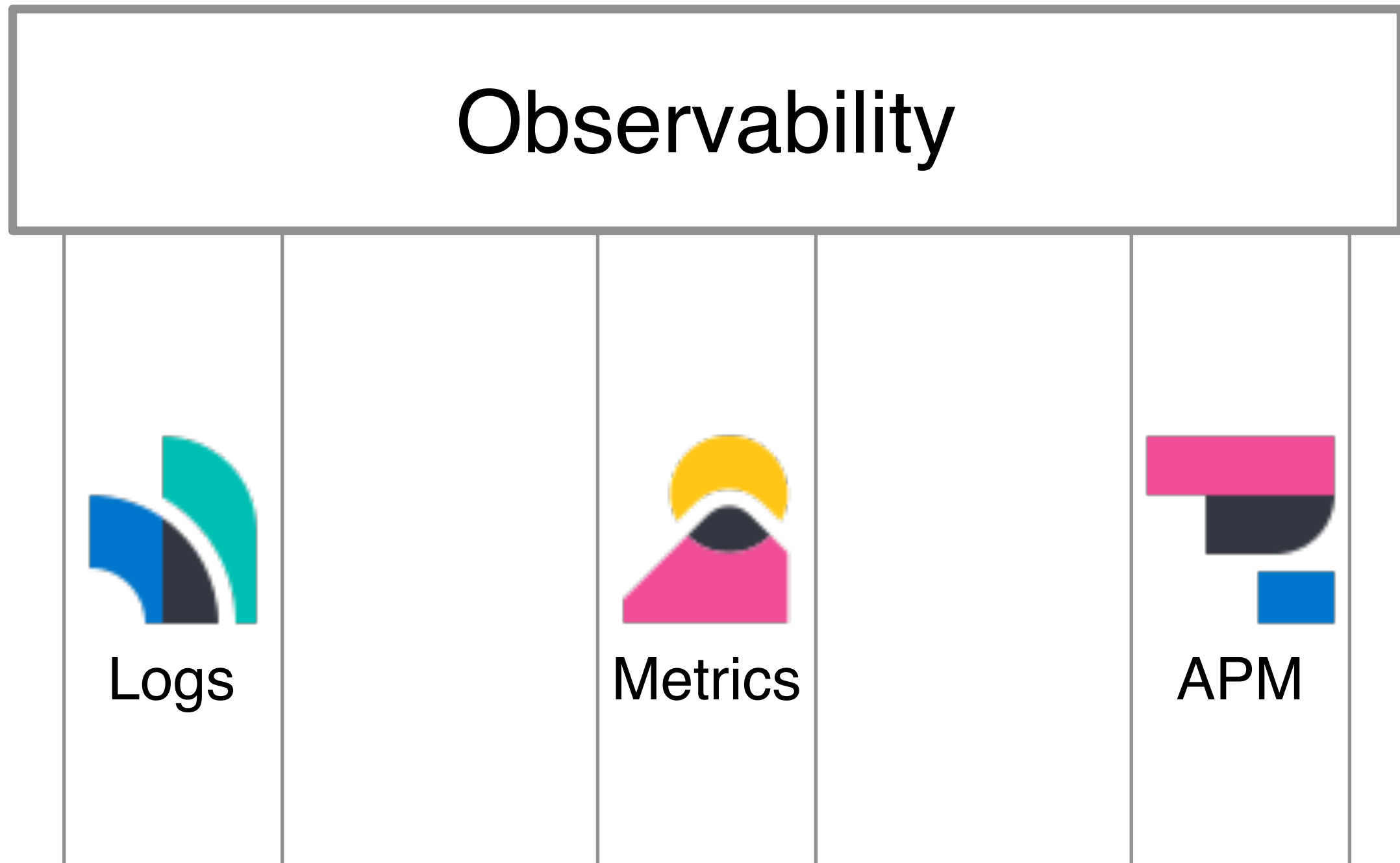
Observability with the Elastic Stack





Observability

- It is not a technology
- It is an attribute of a system
 - like high availability, stability and usability
- It helps detect undesirable behaviors
 - e.g. errors, service downtime and slow responses
- It provides granular information to debug production issues quickly and efficiently
 - e.g. application traces, event logs and resource information

The Pillars of Observability



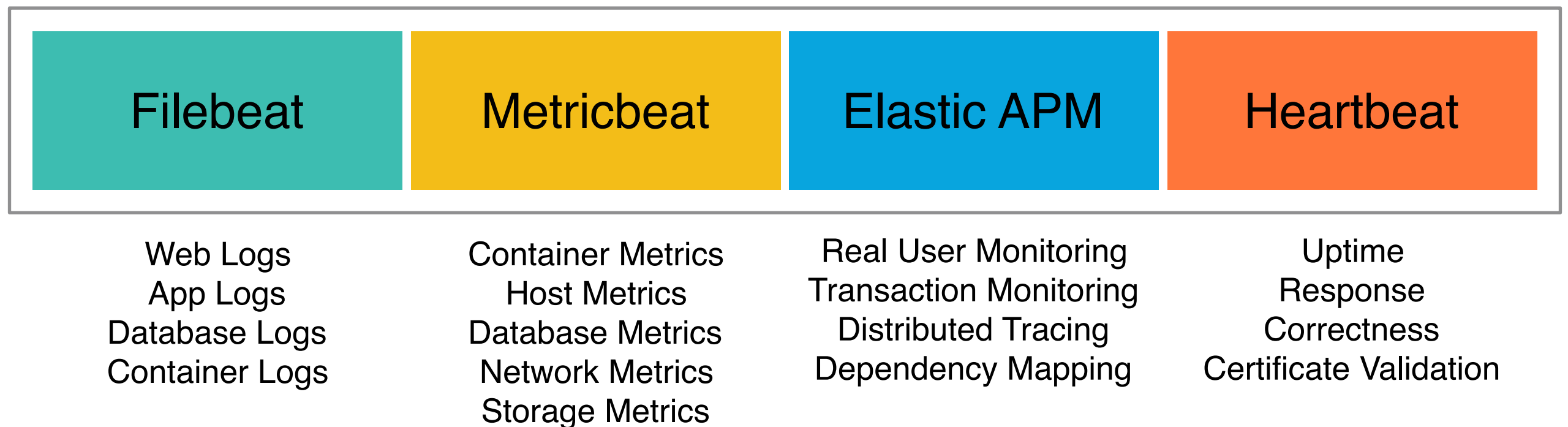
Extra Pillars of Observability

- Uptime Monitoring 
 - it backs to old school monitoring
 - but it still provides important data for observability
 - because it indicates how often a service is available
- Machine Learning 
 - only collecting data is not enough
 - data should be actionable
 - ML detects and alerts about anomalies in observability data

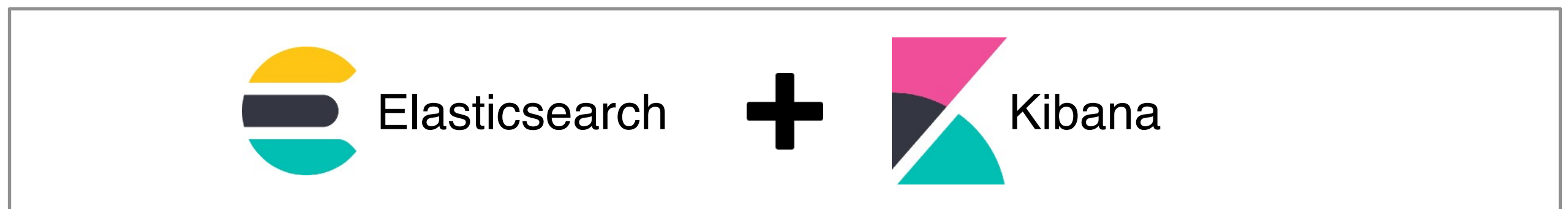
Observability with the Elastic Stack

- With different vendors observability data get isolated
- With the Elastic Stack you can unify your observability data

Dev & Ops Teams



Elastic Common Schema



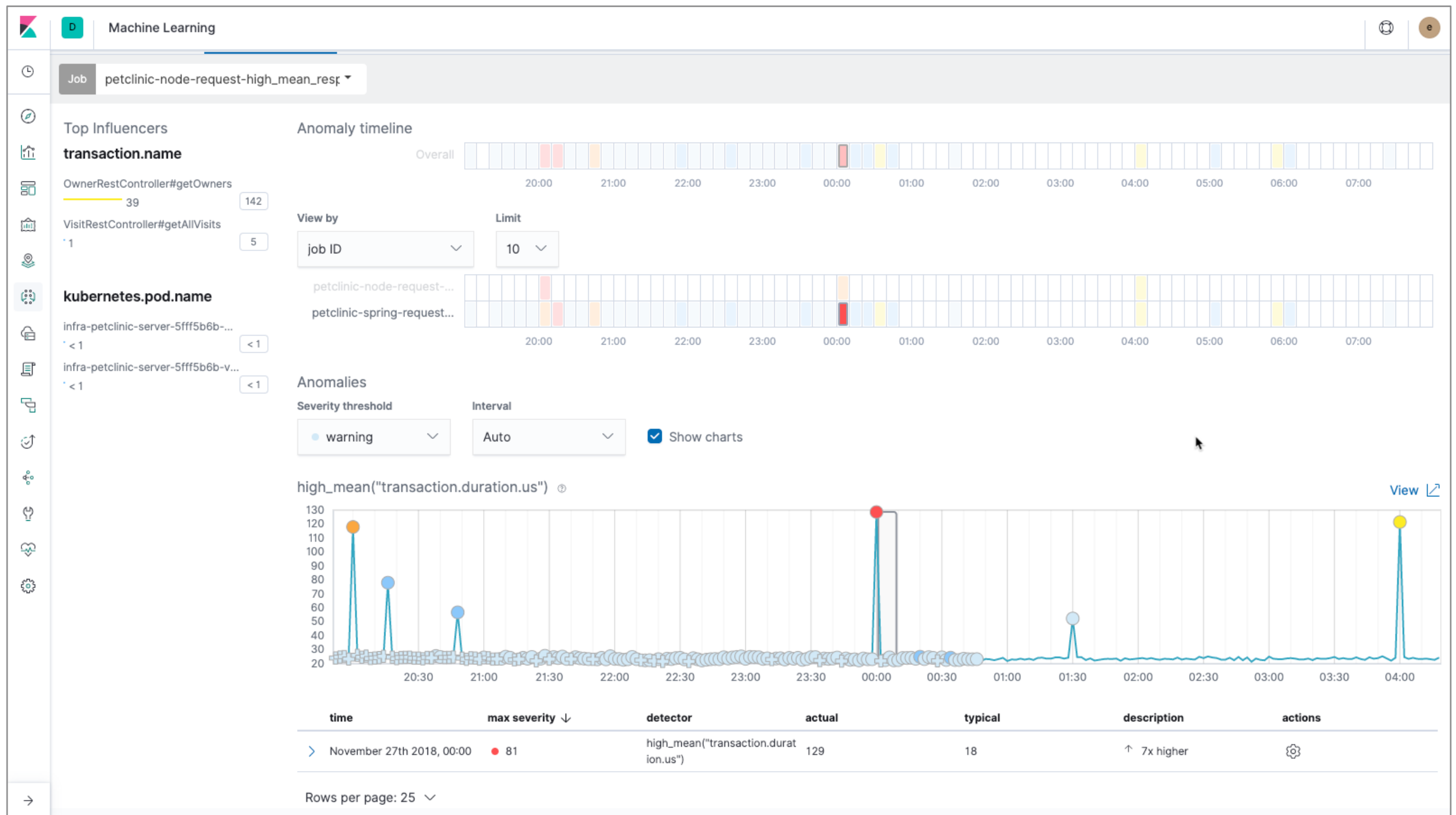
Unified User Interface

- Same UI for KPI summaries and root cause analysis



Unified Machine Learning

- Correlate multiple data sources for better anomaly detection

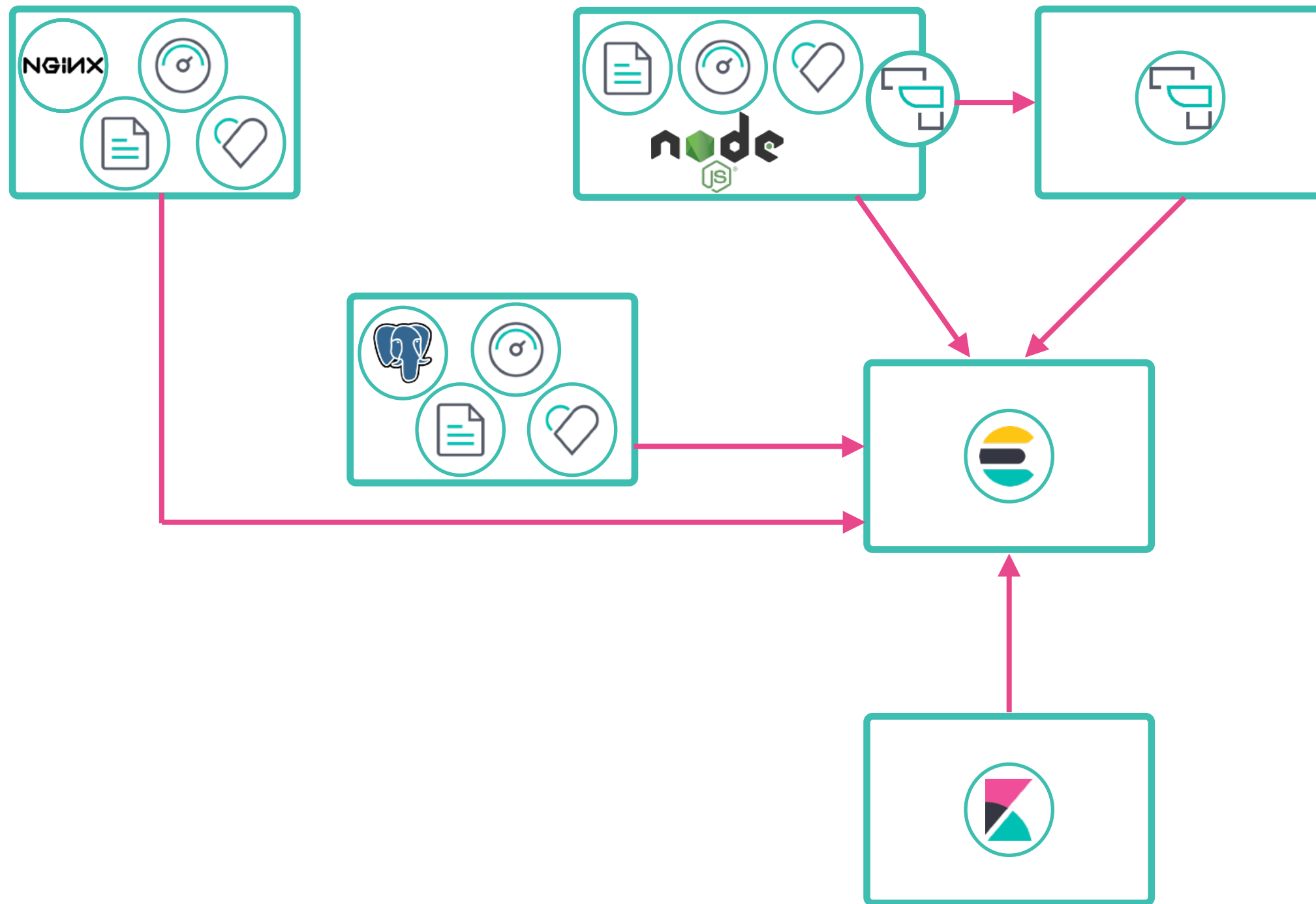


Unified Alerting

- Trigger off any operational data to provide unified SLA monitoring

The screenshot displays the Elastic Alerting interface for creating a new threshold alert. The left sidebar shows the navigation menu with categories: Elasticsearch (Index Management, Index Lifecycle Policies, Rollup Jobs, Watcher, 8.0 Upgrade Assistant), Kibana (Index Patterns, Saved Objects, Spaces, Reporting, Advanced Settings), Logstash (Pipelines), Beats (Central Management), and Security (Users, Roles). The main panel is titled "Create a new threshold alert" with the subtitle "Send an alert when a specific condition is met. This will run every 1 minute." The "Name" field is set to "Response Sizes Large". The "Indices to query" field is set to "apm-7.0.0-span". The "Time field" dropdown menu is open, showing options: "date", "@timestamp" (selected), and "event.created". The "Run watch every" field is set to "1" minutes. Below the configuration fields, a section titled "Matching the following condition" displays a query: "WHEN max() OF node_stats.os.cpu.load_average.1m GROUPED OVER top 20 'source_node.name' IS ABOVE 10 FOR THE LAST 100 minutes". A line graph below the query shows the CPU load average over time, with a red horizontal line at 10 indicating the threshold. The graph shows several peaks, with the highest peak reaching 10 at approximately 09:00.

Observability Data Flow



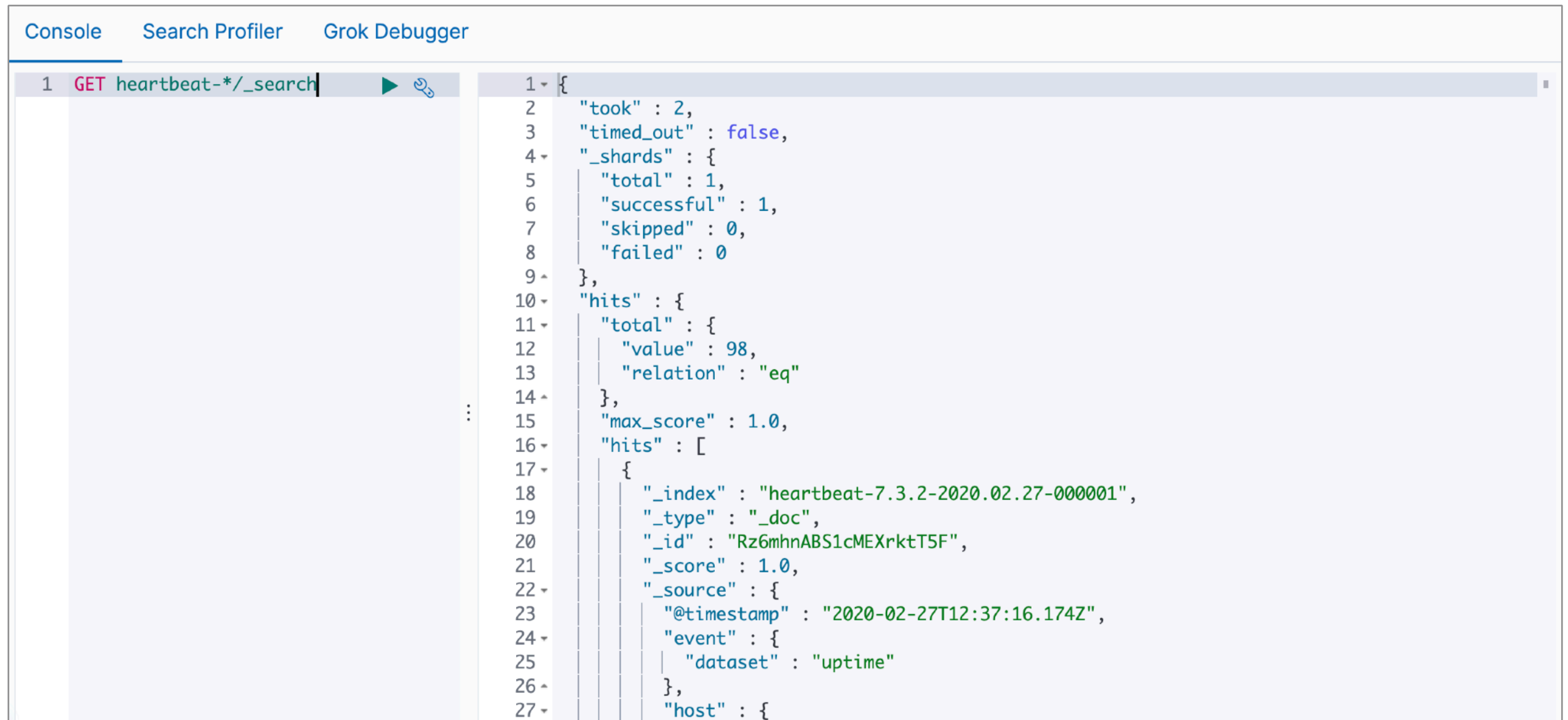
Elasticsearch Indices

- Once the data starts flowing from Beats to Elasticsearch, it is possible to look at the data
- By default, all Beats group data by day using the format:
 - {type}beat-{version}-{yyyy-MM-dd}-XXXXXX

Console		Search Profiler	Grok Debugger
1	GET _cat/indices		
1	green	open	.kibana_task_manager
2	yellow	open	heartbeat-7.3.2-2020.02.27-000001
3	green	open	.kibana_1
4			

Data in Elasticsearch

- To see what the data looks like, it is possible to send a query to Elasticsearch

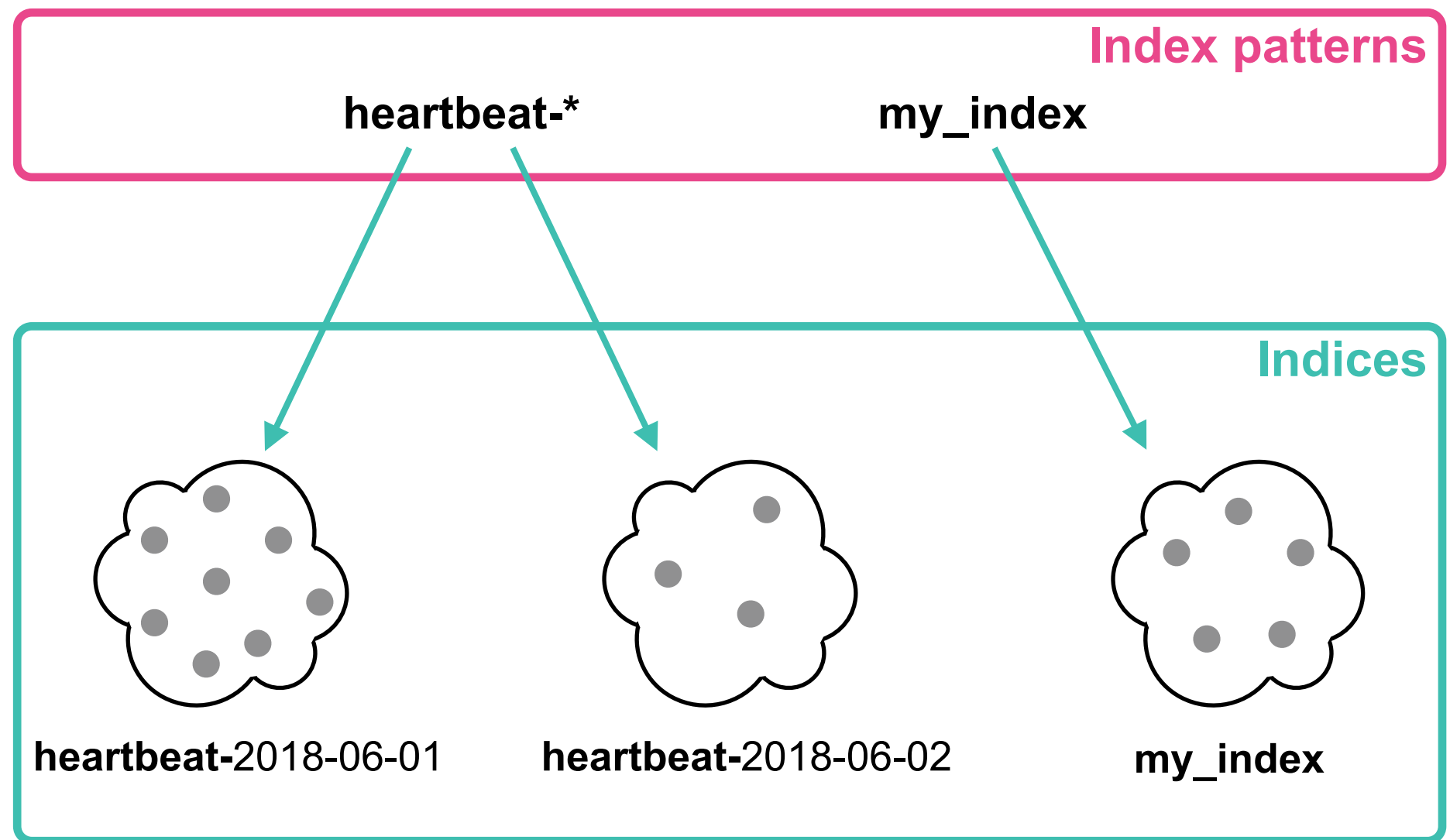


The screenshot shows the Elasticsearch console interface. At the top, there are tabs for 'Console', 'Search Profiler', and 'Grok Debugger'. The 'Console' tab is active, displaying a search query and its results. The query is a GET request to the `heartbeat-*/_search` endpoint. The results are a JSON object containing search statistics and a list of hits. The first hit is a document from the `heartbeat-7.3.2-2020.02.27-000001` index, with a score of 1.0. The document contains fields for `_index`, `_type`, `_id`, `_score`, `_source`, and `host`. The `_source` field contains a timestamp and an event object with a dataset of 'uptime'.

```
1 GET heartbeat-*/_search
2 {
3   "took" : 2,
4   "timed_out" : false,
5   "_shards" : {
6     "total" : 1,
7     "successful" : 1,
8     "skipped" : 0,
9     "failed" : 0
10  },
11  "hits" : {
12    "total" : {
13      "value" : 98,
14      "relation" : "eq"
15    },
16    "max_score" : 1.0,
17    "hits" : [
18      {
19        "_index" : "heartbeat-7.3.2-2020.02.27-000001",
20        "_type" : "_doc",
21        "_id" : "Rz6mhnABS1cMEXrktT5F",
22        "_score" : 1.0,
23        "_source" : {
24          "@timestamp" : "2020-02-27T12:37:16.174Z",
25          "event" : {
26            "dataset" : "uptime"
27          }
28        },
29        "host" : {
```

Indices and Index Pattern

- Unified UI is nice, but **which dataset** is being used?
 - when you search or analyze data, you do that on top of a dataset



Index Patterns

★ heartbeat-*

★

↺

🗑️

Time Filter field name: @timestamp

Default

This page lists every field in the **heartbeat-*** index and the field's associated core type as recorded by Elasticsearch. To change a field type, use the Elasticsearch [Mapping API](#) 📄

Fields (331)

Scripted fields (0)

Source filters (0)

🔍 Filter

All field types ▾

Name	Type	Format	Searchable	Aggregatable	Excluded
@timestamp 🕒	date		●	●	✎
_id	string		●	●	✎
_index	string		●	●	✎
_score	number				✎
_source	_source				✎
_type	string		●	●	✎
agent.ephemeral_id	string		●	●	✎
agent.hostname	string		●	●	✎
agent.id	string		●	●	✎
agent.name	string		●	●	✎

Rows per page: 10 ▾

< 1 2 3 4 5 ... 34 >

Default Visualizations

- Every Beats come with a set of pre-build visualizations
- Kibana can help visualizing different kinds of observability data based on the service or the system monitored
- There are two main ways to load default visualizations
 - setup command (covered in the labs)
 - configuration file (not covered in this training)

Elastic Common Schema

- Observability data comes from different sources
- Visualizations are built on top of observability data
 - so operators can pin down the root cause of issues
- How to correlate data from different sources?
- ECS is a specification on how data should be formatted before being index into Elasticsearch
- Beats apply this standard automatically
- ECS allows the analysis of data coming from different sources

Lesson 1

Review - Observability with the Elastic Stack



Summary

- Observability is a search use case
- Observability helps detect undesirable behaviors and provides granular information to debug production issues quickly and efficiently
- The three main pillars of observability are logs, metrics and application traces, but uptime and machine learning are extra pillars often present in observable systems
- The Elastic Stack provides a way to have a unified implementation of observability
- Elastic Common Schema is a standard defined by Elastic to make sure that all the data collected from different sources can be correlated

Quiz

1. **True or False:** Observability is just a new name for old school monitoring.
2. Which are the three main pillars of observability?
3. **True or False:** One of the benefits of using the Elastic Stack to implement observability is to have all information in a single operational source with the ability to automatically correlate this data in an intuitive user interface.

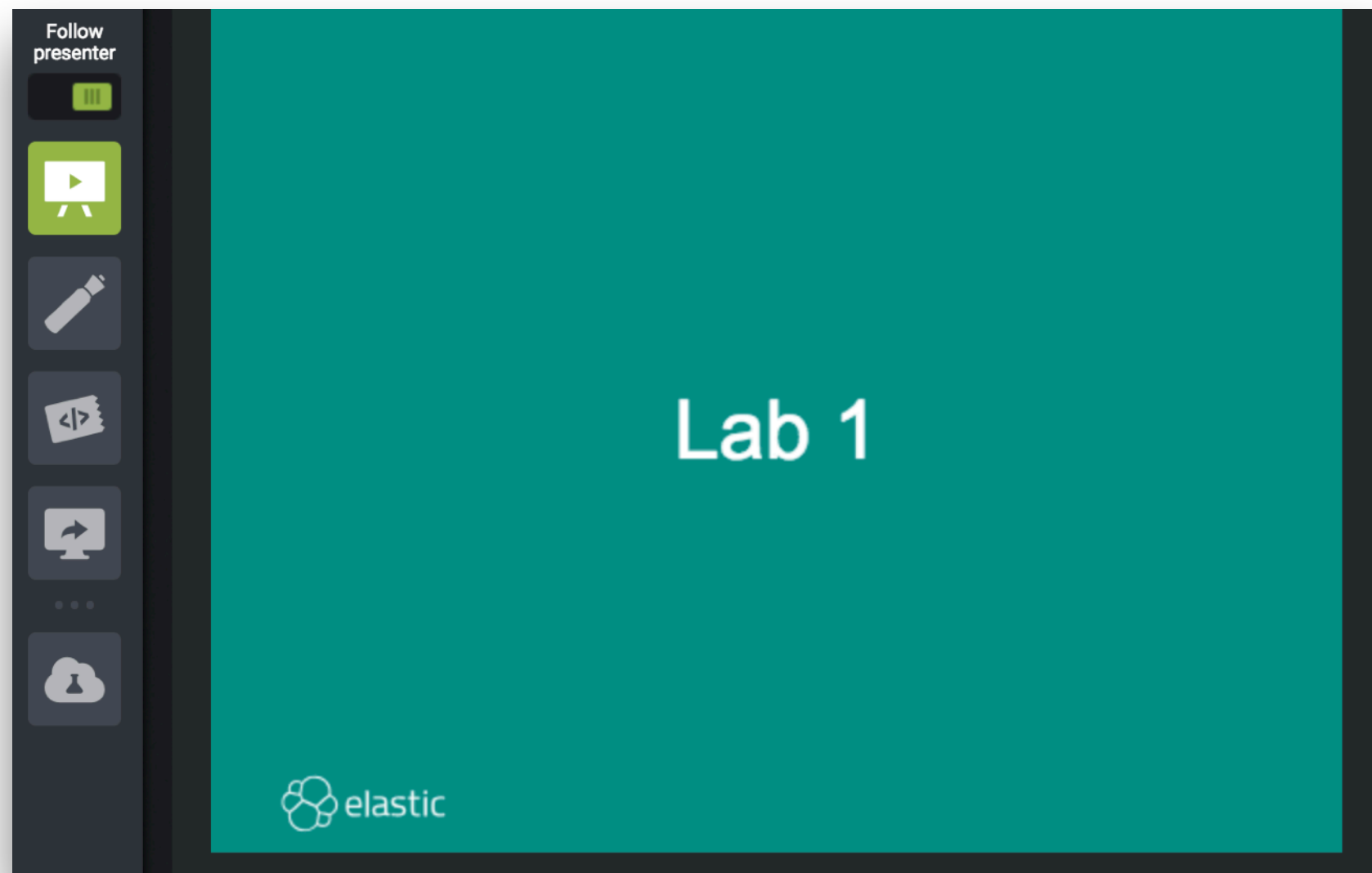
Lesson 1

Lab - Observability with the Elastic Stack



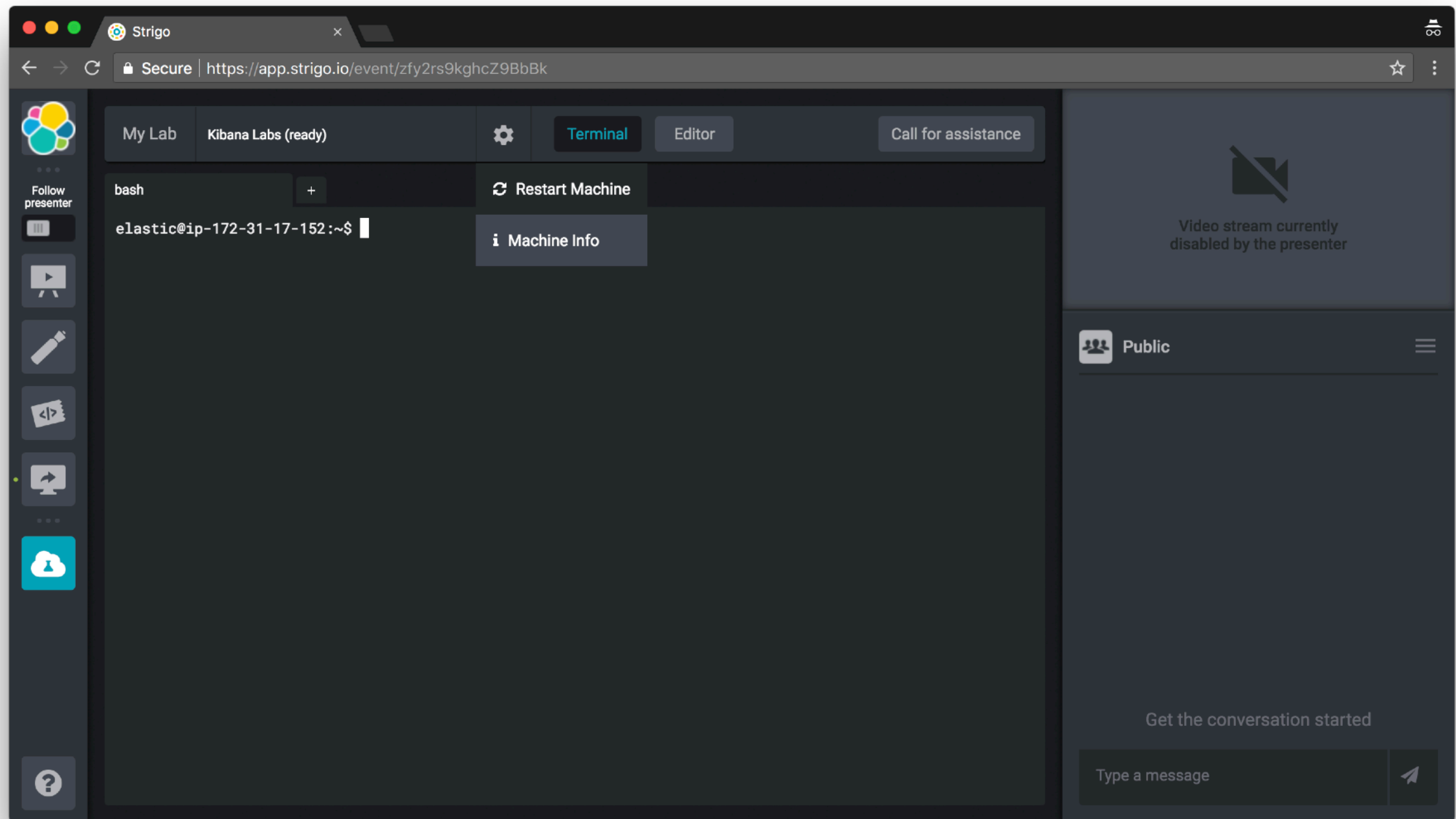
Lab Environment

- Visit Strigo using the link that was shared with you, and log in if you haven't already done so
- Click on "**My Lab**" on the left



Lab Environment

- Click on the gear icon next to "My Lab" and select "Machine Info"



Lab Environment

- From here you can access lab instructions and guides
 - You also have them in your .zip file, but it is easier to access and use the lab instructions from here:



elastic

Welcome to Observability Fundamentals

-
- [Lab Instructions](#)
 - [Virtual Classroom User Guide](#)
 - [Kibana](#)
-

© Elasticsearch BV 2015-2020. All rights reserved. Decompiling, copying, publishing and/or distribution without written consent of Elasticsearch BV is strictly prohibited.

Lesson 2

Logs



Business Questions

How many users visited our new training landing page?

Why is our Javascript application slow?

???

When should we schedule download service maintenance?

How many webinar signups did we get from Europe?



It's all in the logs!

- Jordan Sissel

What is a log?

- logs are **records of activities**
 - by a system
 - by an application
 - by a device
 - by a human
 - ...
- **timestamp** + data

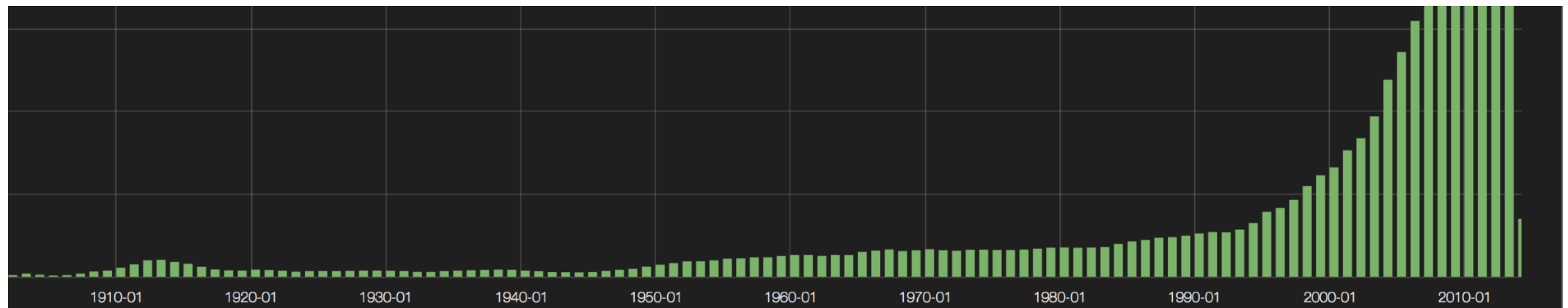
What are Logs?

- Application Logs

```
66.249.73.185 - - [16/Feb/2014:09:47:54 -0500] "GET / HTTP/1.1" 200 37932 "-"  
"Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
```

```
[2017-05-18 00:00:05,871][INFO ][cluster.metadata          ] [esndata-2] [.data-  
es-1-2017.05.18] creating index, cause [auto(bulk api)], templates [.data-  
es-1], shards [1]/[1], mappings [_default_, shards, node, index_stats,  
index_recovery, cluster_state, cluster_stats, node_stats, indices_stats]
```

```
120707 0:37:09 [Note] Plugin 'FEDERATED' is disabled.  
120707 0:37:09 InnoDB: The InnoDB memory heap is disabled
```



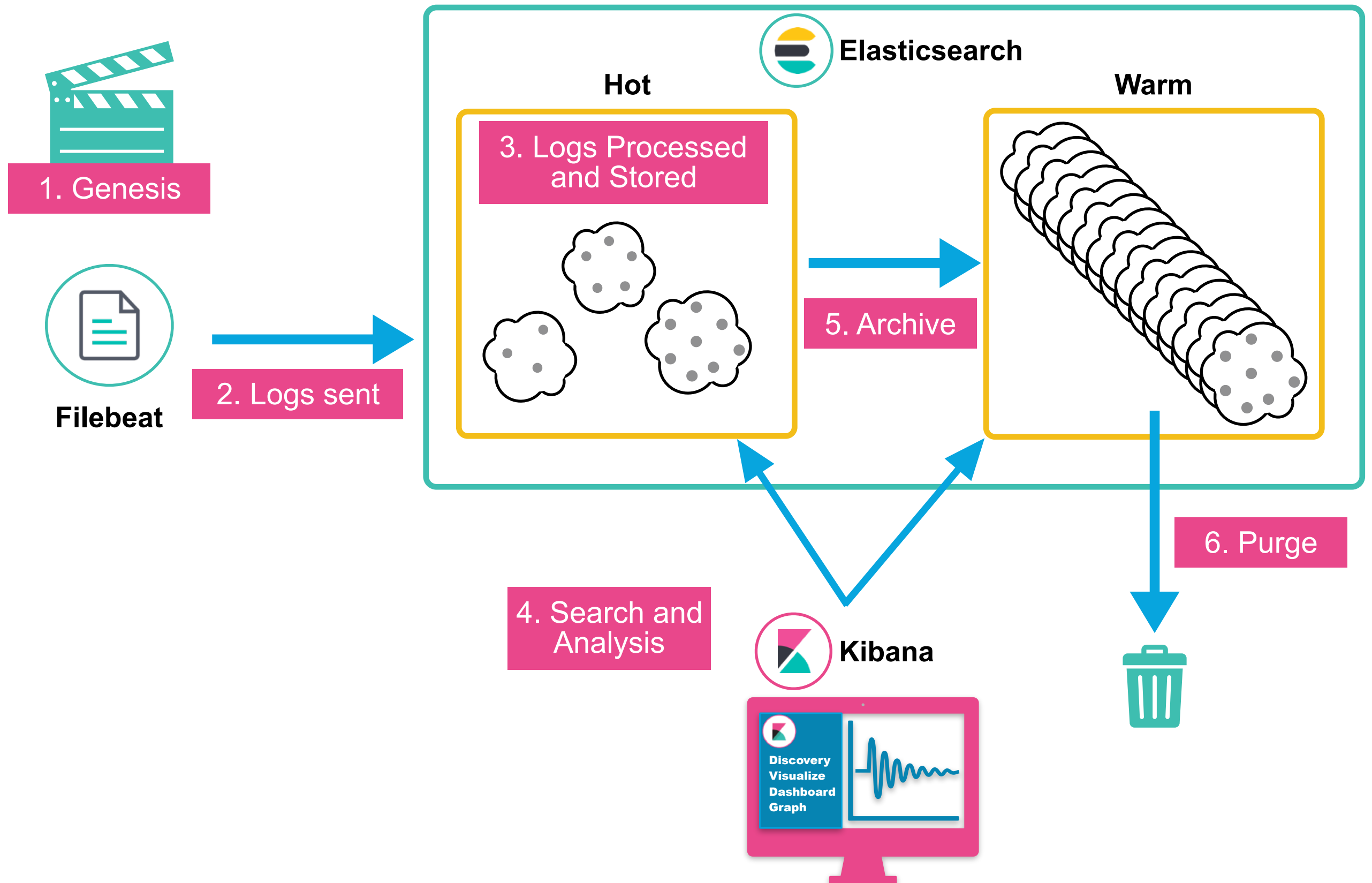
A Note About Timestamp

- Timestamps can be difficult to deal with
 - many different formats
 - time zones can be challenging
- Elasticsearch prefers ***ISO 8601*** format
 - but can ingest multiple time formats, if configured
- The ***ISO 8601*** format was designed to be unambiguous
 - it is a good practice to represent timestamps in ***ISO 8601***
- UI applications (e.g. Kibana) can adjust the displayed time
 - combine the stored time with the user's local time
- Example: **"2018-10-05T14:30:00Z"**

Common Problems

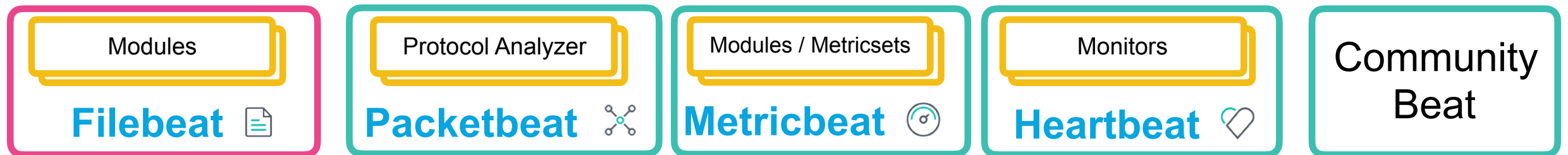
- Consistency
 - every application and device logs in its own format
- Time Formats
 - "Oct 11 20:21:47", "020805 13:51:24"
- Decentralized
 - logs are spread across all of your servers
 - SSH + grep aren't scalable
- Experts Required
 - limited access to log files on servers
 - limited knowledge of the log format

Logs Lifecycle



Beats

- Multiple flavors



- Beats can resolve different issues
 - reading files
 - retrieving metrics
 - retrieving network data
 - testing services availability

Filebeat

- Filebeat Modules
 - simplify the collection, parsing, and visualization of common log formats



Apache



Nginx



MongoDB



MySQL



PostgreSQL



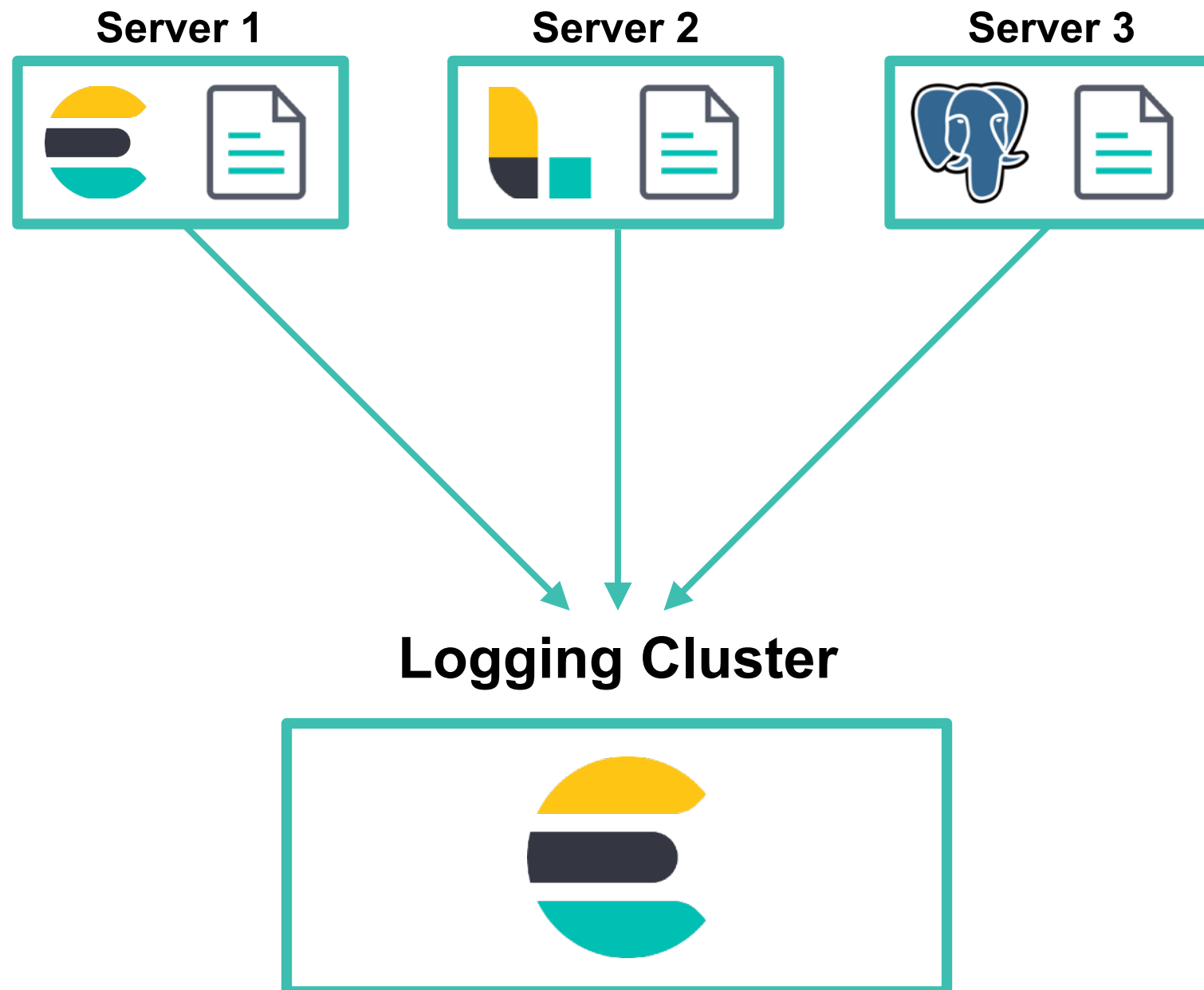
Redis



Add your own

Filebeat

- Open source data shippers
- Lightweight agent collecting logs



Getting Started with Filebeat

1. Download Filebeat on the host where the logs reside
2. Configure Filebeat
3. Start Filebeat
4. Verify that the data has arrived in Elasticsearch



Lesson 2

Review - Logs



Summary

- Logs can give us the answers to many questions which we ask of our data
- A log consists of a message with both a timestamp and some piece(s) of data
- Filebeat monitors log directories or specific log files
- Filebeat Modules simplify the collection, parsing, and visualization of common log formats
- Once the data is sent to Elasticsearch, it is possible to query Elasticsearch to explore the data

Quiz

1. What are the two elements of a log message?
2. **True or False:** It's OK to give Scott from Marketing ssh access to the web server so he can see how many people signed up for his latest webinar.
3. **True or False:** Filebeat Modules simplify the collection, parsing, and visualization of common log formats.

Lesson 2

Lab - Logs



Lesson 3

Metrics

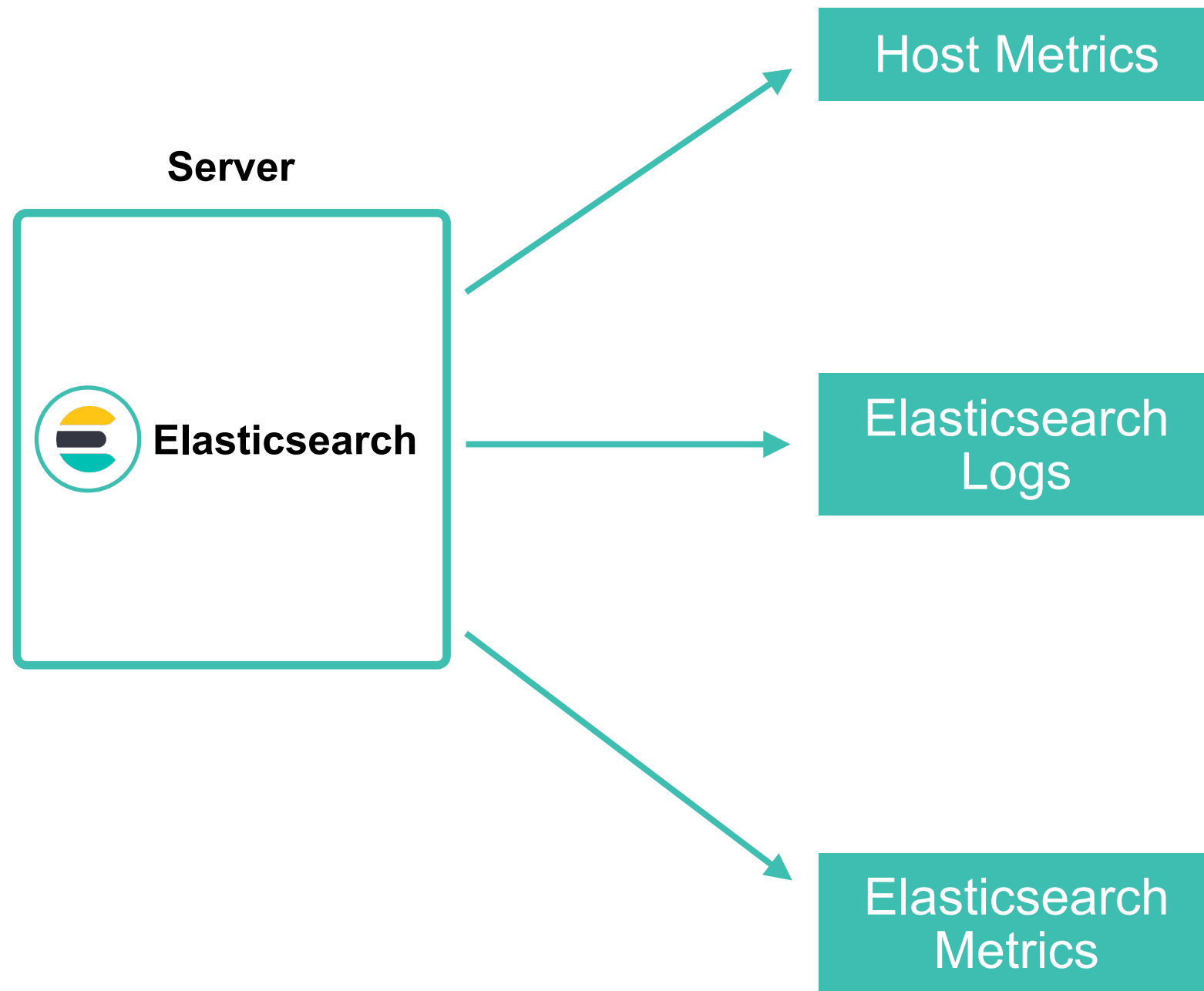


Monitoring

- Systems and services are generating a lot of data that should be:
 - stored
 - analyzed
 - monitored



Monitoring Example



Logs x Metrics

- There are some similarities between logs and metrics
 - they are both time series oriented
 - both can contain keywords
- But they have fundamental differences
 - metrics focus on a **periodic measurement** of a system
 - metrics focus on numerical values and keywords
 - logs describe an event that happened (what and when)
 - logs often contain text that needs to be parsed

Logs x Metrics Example

Logs are recorded
as events occur

[2018-09-07T07:48:00,127][INFO][o.e.x.m.MIDailyMaintenanceService] triggering scheduled [ML] maintenance tasks

[2018-09-07T07:48:00,381][INFO][o.e.x.m.a.TransportDeleteExpiredDataAction] [_8LMCWq] Deleting expired data

[2018-09-07T07:48:00,648][INFO][o.e.x.m.MIDailyMaintenanceService] Successfully completed [ML] maintenance tasks

Metrics are recorded
based on a schedule

[2018-09-07T06:00:00,000][filesystem] 50085941248 overlay / 67371577344

[2018-09-07T06:05:00,000][filesystem] 50085917352 overlay / 67371577344

[2018-09-07T06:10:00,000][filesystem] 50075903715 overlay / 67371577344

Timestamps

- Elasticsearch prefers the ISO 8601 format
- The timezone is included in the timestamp
- Kibana can adapt to the local timezone of the user

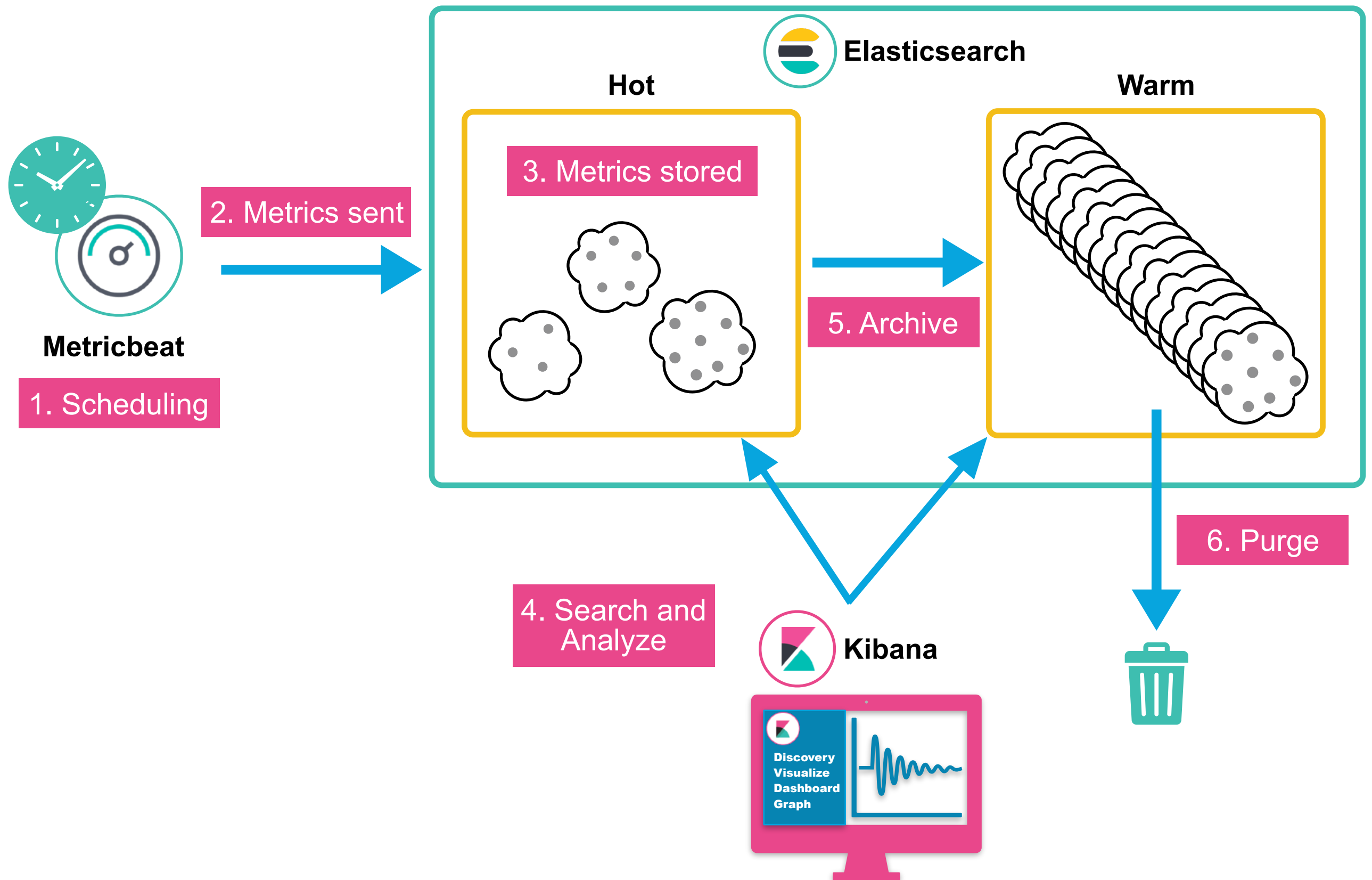
Which
time zone are
we speaking
about?

2018-09-07T06:10:00

2018-09-07 06:10:00 -0400

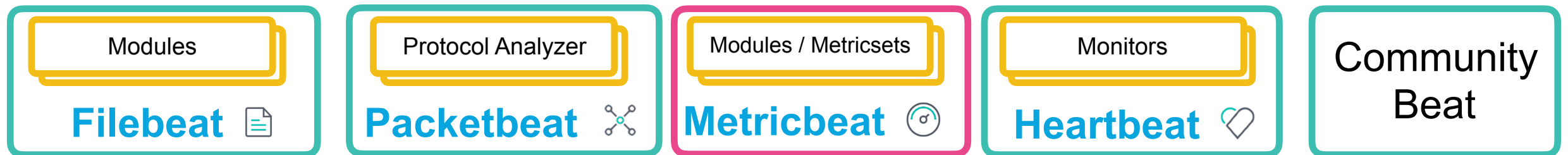
Oh ok! New
York time zone!

Metrics Lifecycle



Beats

- Multiple flavors



- Beats can resolve different issues
 - reading files
 - retrieving metrics
 - retrieving network data
 - testing services availability

Metricbeat

- Collects metrics from the operating system and from services running on the server



System module



Nginx



MongoDB



MySQL



PostgreSQL



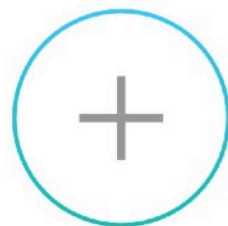
Redis



ZooKeeper

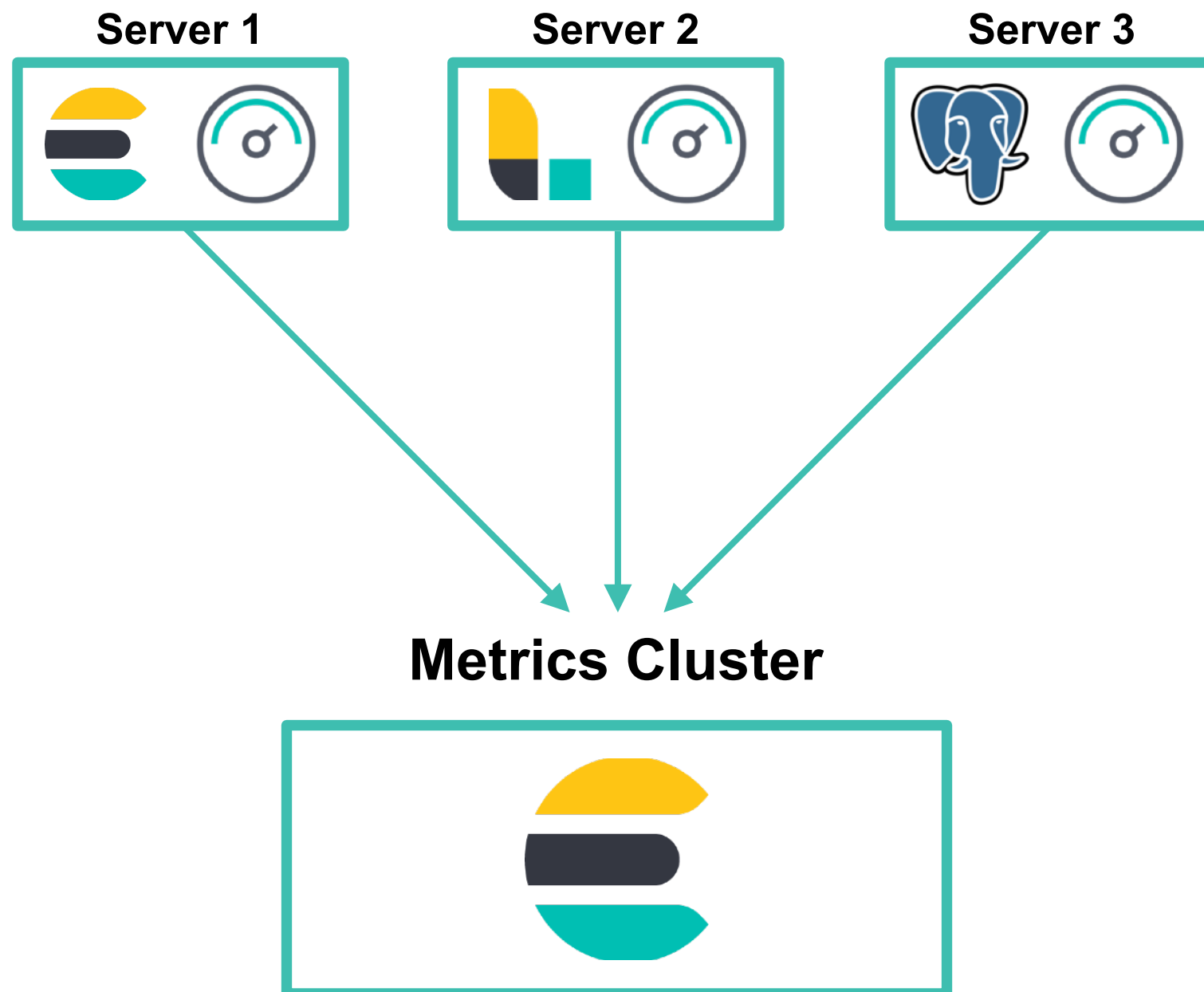


Apache



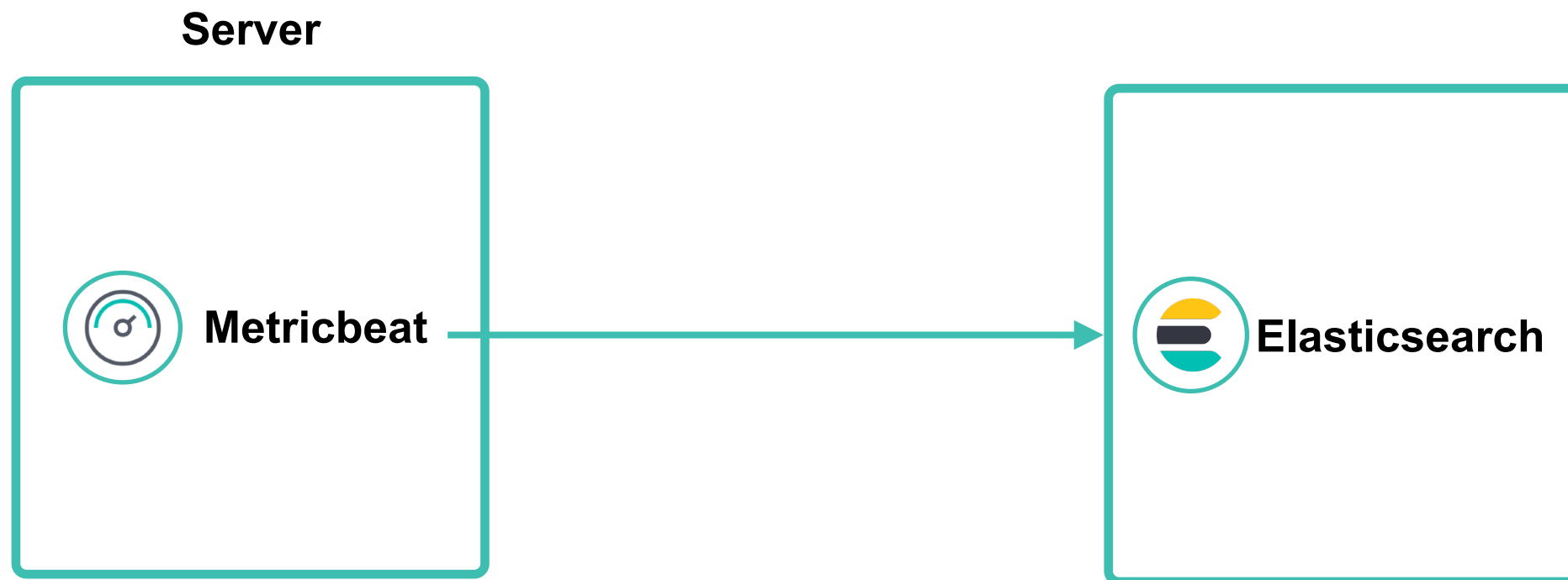
Metricbeat

- Open source data shippers
- Lightweight agent collecting metrics



Getting Started with Metricbeat

1. Download Metricbeat on the host you want to monitor
2. Configure Metricbeat
3. Start Metricbeat
4. Verify that the data is in Elasticsearch



Lesson 3

Review - Metrics



Summary

- Metrics and Logs provide important observability data
- Logs are about what happened and when it happened
- Metrics are about collecting a certain information periodically
- Metricbeat can collect multiple metrics from systems and services
- Once the data is sent to Elasticsearch, it is possible to query Elasticsearch to explore the data

Quiz

1. **True or False:** Metrics only contain numeric values.
2. What is the main difference between logs and metrics?
3. **True or False:** It is better to use timestamps that follow the ISO 8601 format with timezone information when implementing Observability with the Elastic Stack.

Lesson 3

Lab - Metrics



Lesson 4

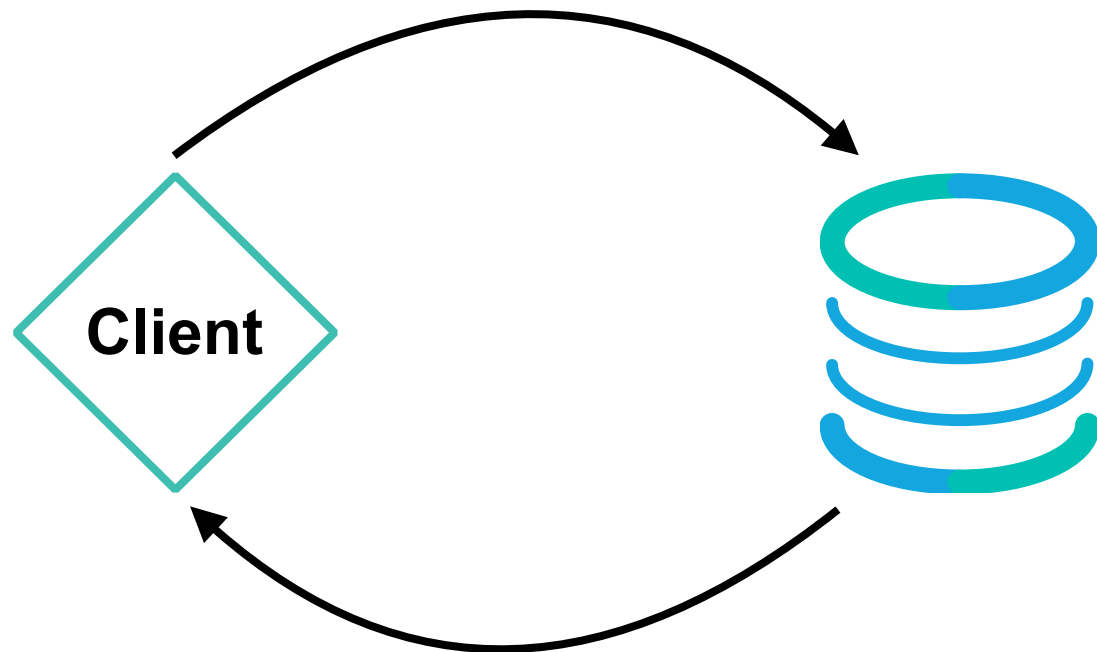
APM



What is APM?

- Application Performance Monitoring

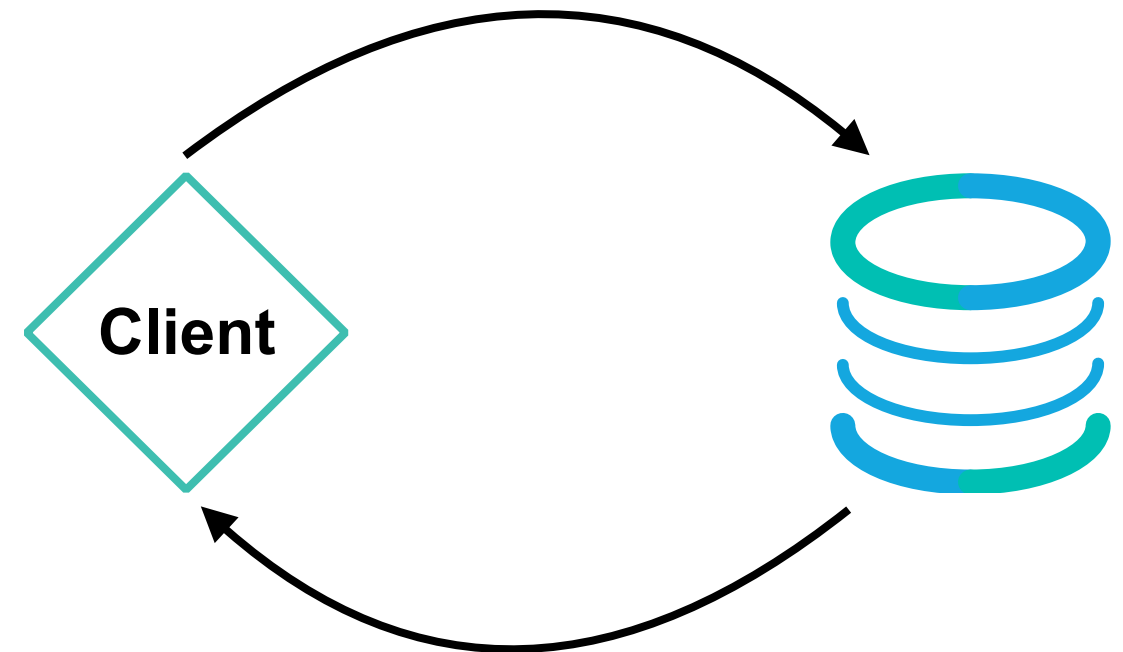
17:36:30 Request /top/10



17:36:38 Response /top/10 200 OK

*Why did my
request take eight
seconds?*

17:36:30 Request /top/10



17:36:31 Response /top/10 500 ERROR

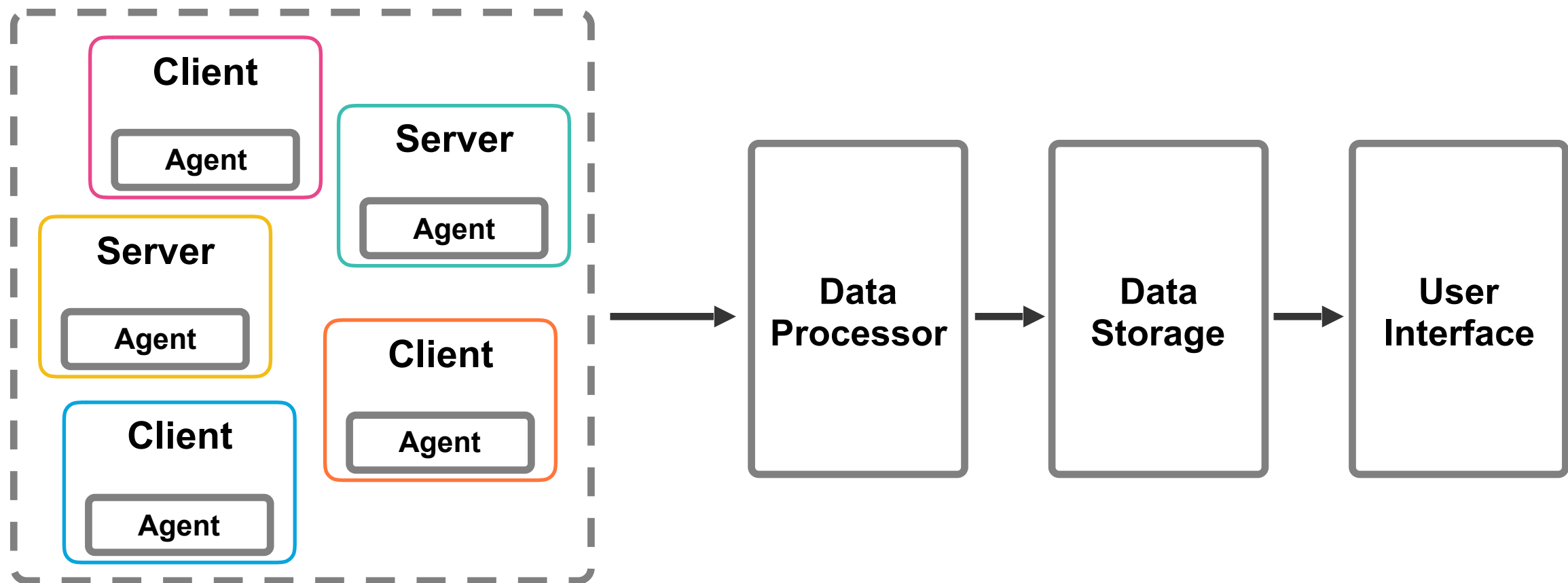
*Why did my
response return a 500
error?*

Why APM?

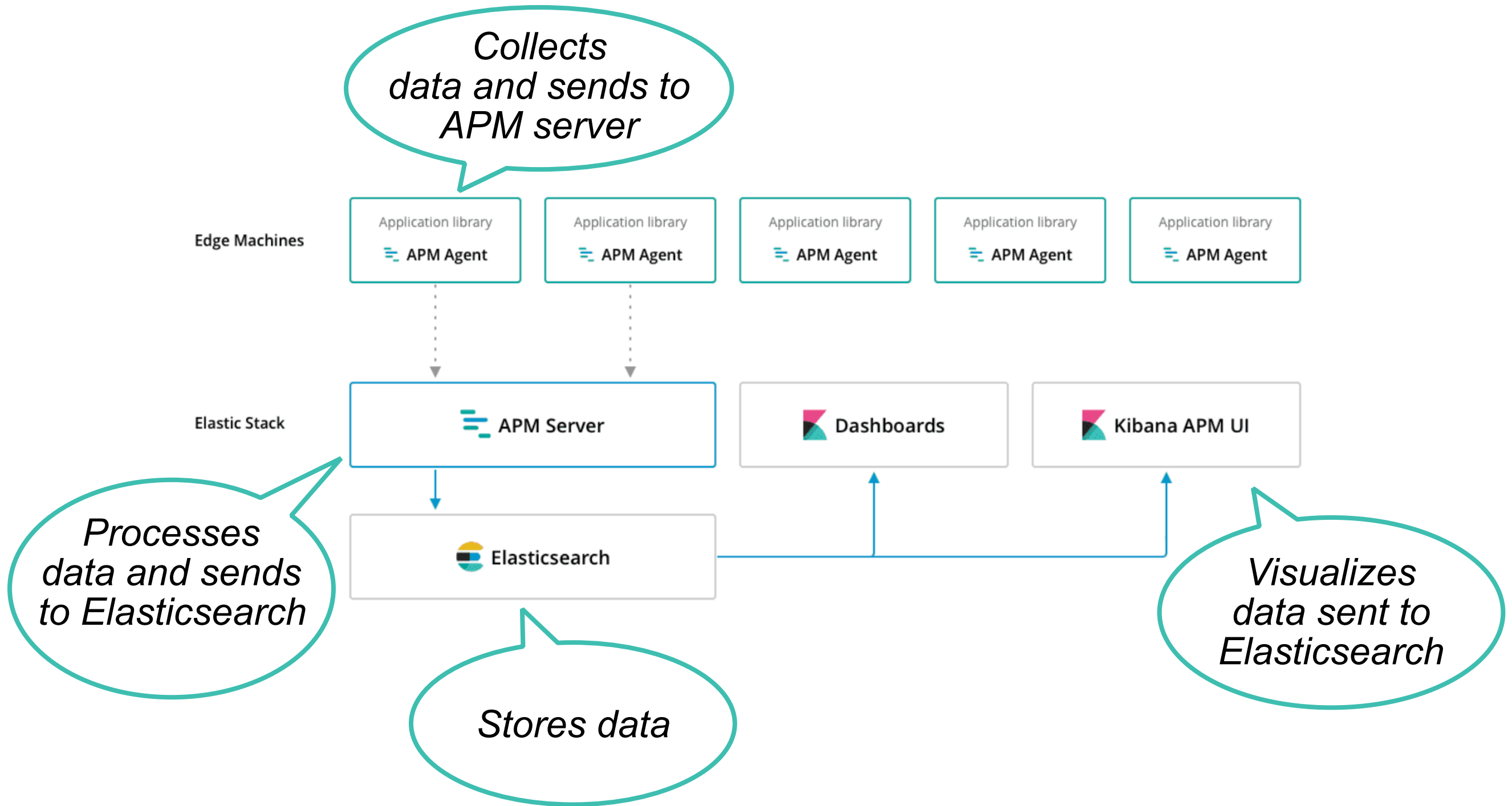
- It records traces for database queries, external HTTP requests, and other slow operations that happen during requests to your application
 - it is easier to see where your application is spending time
- It also collects errors and exceptions that are not handled by your application
 - it is easier to debug errors that might happen in your application
- You can find performance bottlenecks and errors before your customers face them
- You can increase the productivity of your development team

How APM Works?

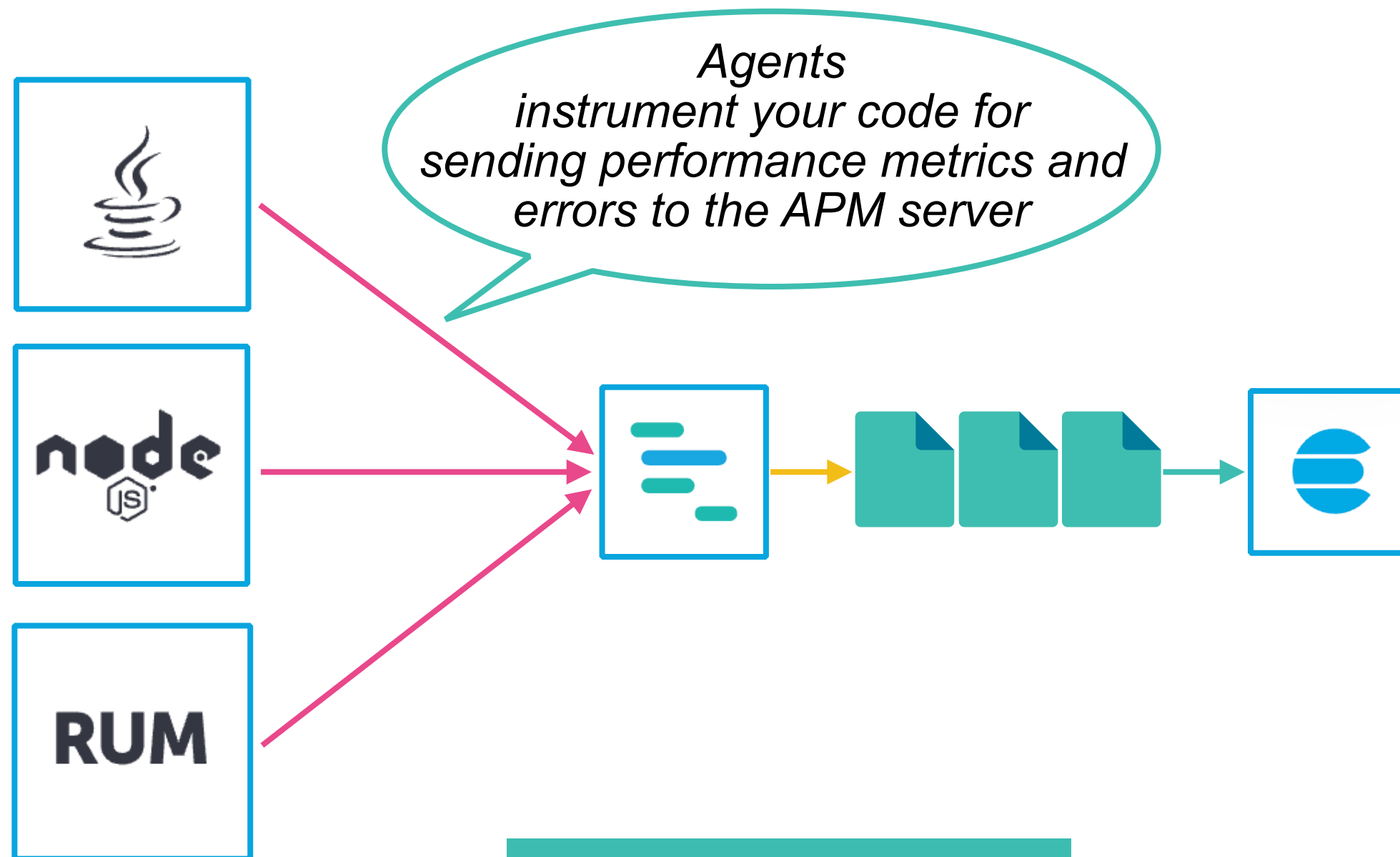
- This is how Application Performance Monitoring works in general



Elastic APM Components



APM Agents Overview



APM agents are written in the same language as your service

APM agents usually have automatic instrumentation for the most popular frameworks and routers

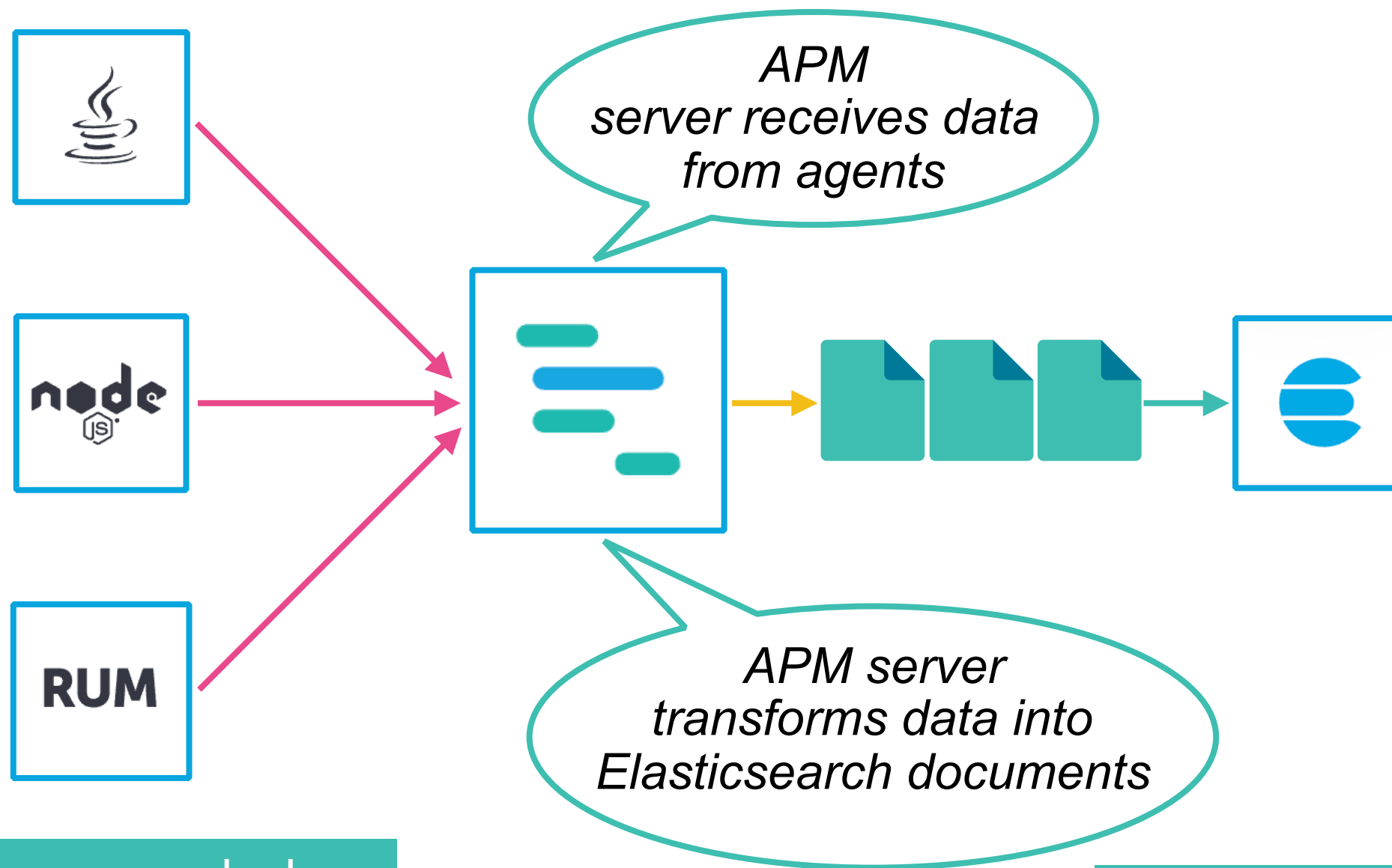
APM agents also allow you to manually instrument your application

Available Agents



These are the Elastic agents,
but there are community
agents too.

APM Server Overview



APM server works by exposing an HTTP server to which agents ship the APM data they collect

APM server is built with the Beats framework, and as such it leverages its functionality

Why is APM Server a Separate Component?

- Performance
 - it can be scaled independently since it is a stateless component
 - it controls the amount of data flowing into Elasticsearch
 - it can buffer data when Elasticsearch becomes unresponsive
- Security
 - it prevents browsers from interacting with Elasticsearch
- Interoperability
 - it acts as a middleware for JavaScript source mapping
 - it provides a JSON API for agents

Data Model

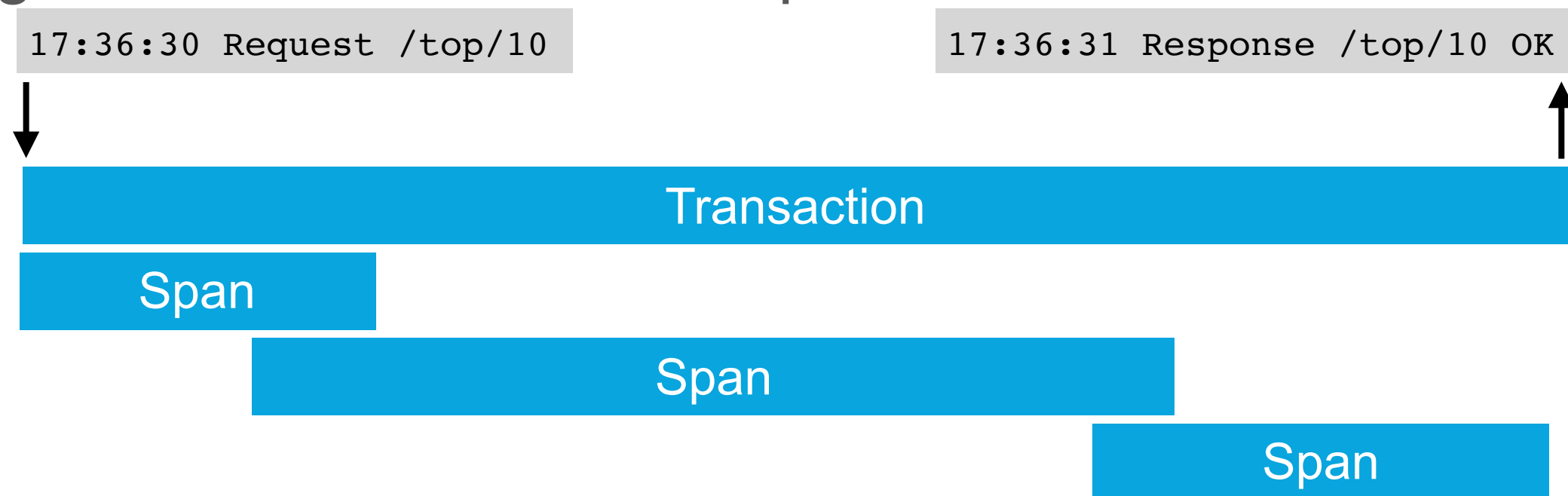
- What kind of data agents send to the APM server?
 - spans, transactions, errors or metrics
 - which are known as events
 - and can contain additional metadata
 - such as labels, custom context and user context

Spans

- Spans contain information about a specific code path that has been executed
- They measure from the start to end of an activity
 - and they can have a parent/child relationship with other spans
- A span contains
 - a transaction id attribute that refers to their parent transaction
 - a parent id attribute that refers to their parent span or transaction
 - start time and duration
 - name
 - type
 - stack trace (optional)

Transactions

- Transactions are a special kind of span that have additional attributes associated with them
- They describe an event captured by an Elastic APM agent instrumenting an application
 - e.g., a HTTP request or an asynchronous background job
 - it includes the timestamp of the event, a unique id, type, name, data about the environment and other relevant information
- Together, transactions and spans form a trace



Errors

- An error is either a captured **exception** or a captured **log**
 - it can contain a **stack trace**, which is helpful for debugging
 - the **culprit** of the error indicating where it originated
 - and might relate to the transaction during which it happened
- For simplicity, errors are represented by a unique ID

```
server/top.js in <anonymous> at line 27
```

```
25.  
26. app.get('/top/10', function (req, res) {  
27.   apm.captureError('this is a string', function (err) {  
28.     if (err) {  
29.       res.status(500).send('could not capture error: ' + err.message)
```

```
server.js in <anonymous> at line 75
```

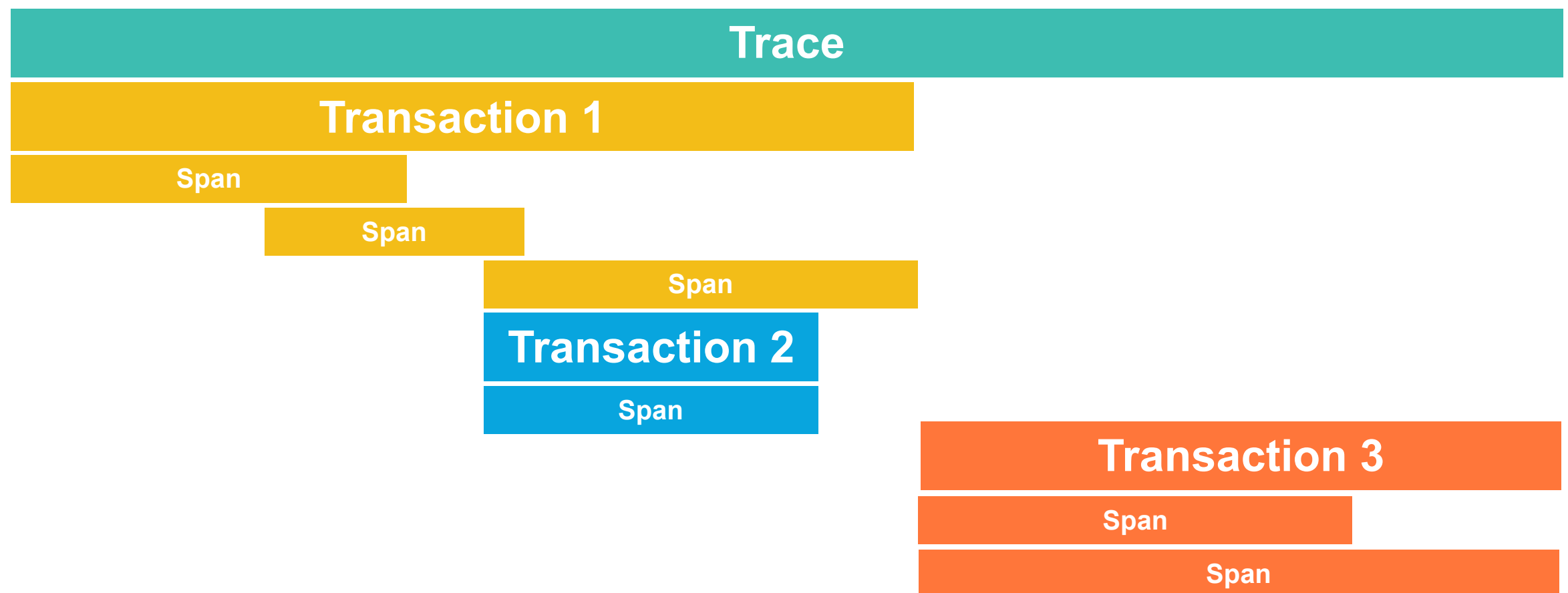
```
73. })  
74.  
75. next()  
76. })  
77.
```

Metrics

- APM agents automatically pick up basic host-level metrics
 - including system and process-level CPU and memory metrics
- Agents specific metrics are also available
 - like JVM metrics in the Java agent
 - and Go runtime metrics in the Go agent

Distributed Tracing

- One trace that spans multiple transactions
- APM does that by giving each trace a unique ID
 - it uses OpenTracing IDs
 - and passes that ID on through the transactions
 - thereby creating an end-to-end trace for multiple applications



Logs x Metrics x APM

- Full-stack monitoring with Elastic APM
- Adds end-user experience and application-level monitoring to the Elastic Stack



Real User Monitoring (RUM)



Application-level monitoring



Server-level monitoring



Logging

Getting Started with Elastic APM

1. Start an Elasticsearch cluster with Kibana (version > 5.6)
2. Start APM server
3. Setup APM agents in your applications
4. Verify that the data is in Elasticsearch

Lesson 4

Review - APM



Summary

- Elastic APM allows you to monitor software services and applications in real time, collecting detailed performance information on response time for incoming requests, database queries, calls to caches, external HTTP requests, etc
- Elastic APM also automatically collects unhandled errors and exceptions
- Elastic APM consists of four components: Elasticsearch, APM agents, APM server, and Kibana APM UI
- Distributed Tracing enables you to analyze performance throughout your micro services architecture all in one view
- Once the data is sent to Elasticsearch, it is possible to query Elasticsearch to explore the data

Quiz

1. How Elastic APM helps monitoring your application?
2. How Elastic APM helps debugging your application?
3. What are the four components of Elastic APM?

Lesson 4

Lab - APM



Quiz Answers



Observability with the Elastic Stack

1. False. Observability is more than just detecting undesirable behaviors and it is also about providing operators with granular information to debug production issues quickly and efficiently.
2. Logs, Metrics and APM.
3. True. Implementing observability with the Elastic Stack allows the correlation of data through unified UIs, machine learning and alerting.

Logs

1. Timestamp + Data
2. False. A better option is to give Scott access to Kibana, so he can run queries and check visualizations on log data.
3. True. Filebeat includes many modules to make it easier parsing and visualizing logs from common formats.

Metrics

1. False. They can also contain keywords.
2. Logs are about what happened and when it happened, while metrics are about collecting the same information periodically.
3. Elasticsearch prefers the ISO 8601 format for timestamps with the timezone included, so Kibana can adapt to the local timezone of the user.

APM

1. By collecting detailed performance information about requests to your application
2. By collecting unhandled errors and exceptions
3. APM agents, APM server, Elasticsearch and Kibana APM UI