

EPL660: Information Retrieval and Search Engines – Lab 7



**University of Cyprus
Department of
Computer Science**

Παύλος Αντωνίου

Γραφείο: B109, ΘΕΕΕ01

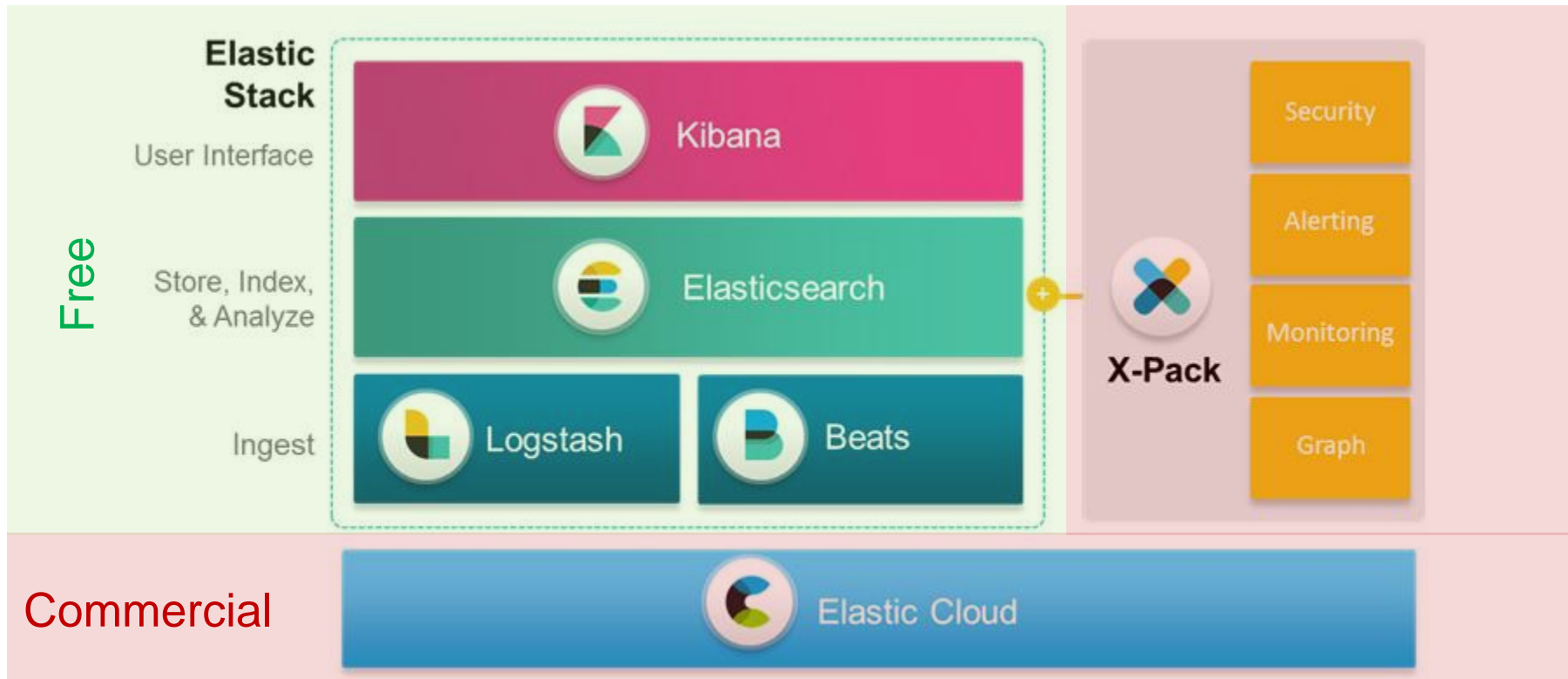


- Free, open-source, document (json) oriented **search and analytics engine** built on top of Apache Lucene
- Developed in Java (cross-platform)
- Distributed (cloud-based)
- Scalable and highly available
- Java API + RESTful HTTP/JSON API
- Used for full-text search, structured search, analytics, or all three in combination
- Near Real-Time searching
 - slight latency (~ 1 sec) from the time you index a document until the time it becomes searchable

Elastic Stack








elastic



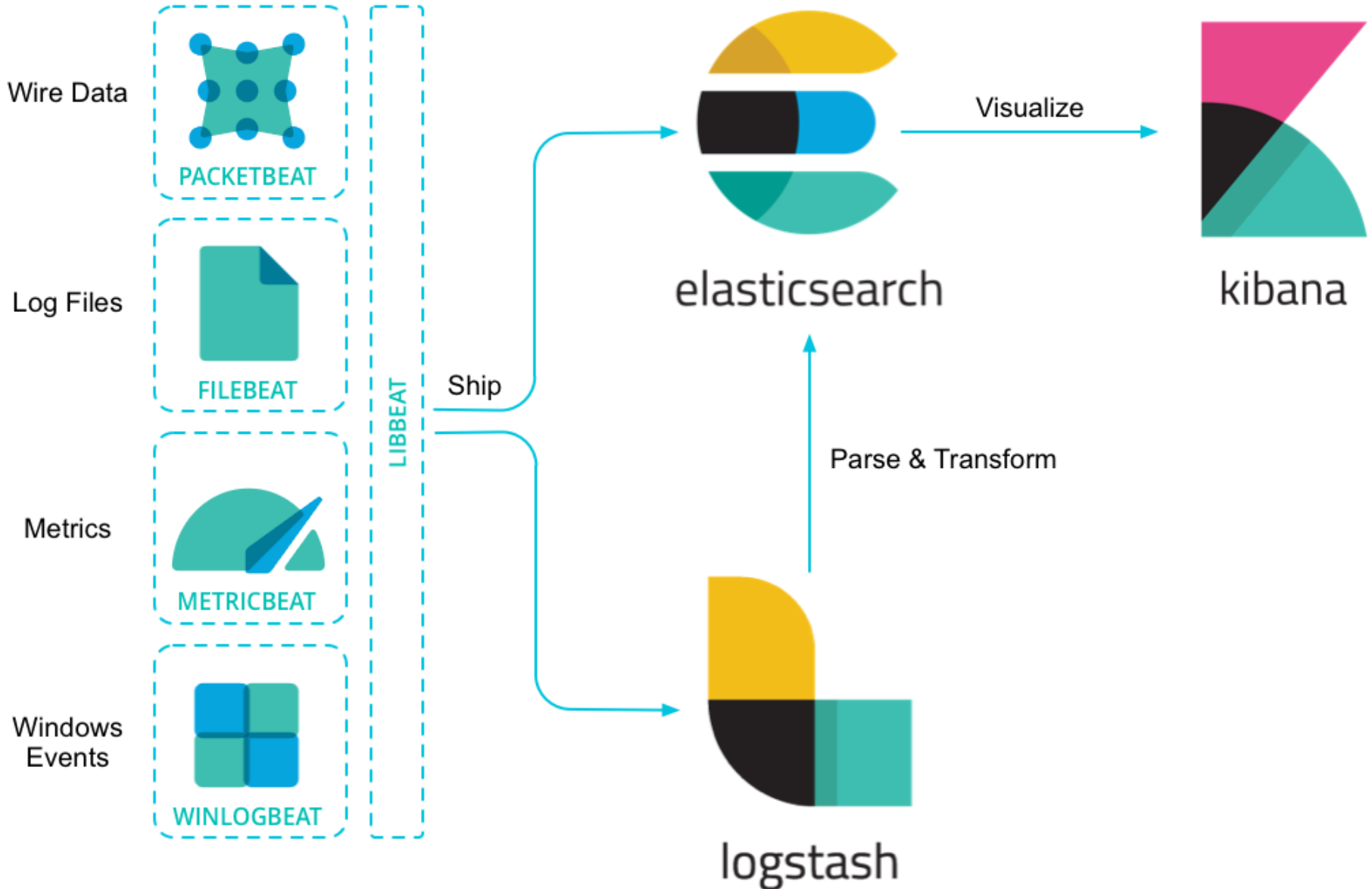
Elasticsearch is a search and analytics engine.

Rest Elastic Stack components



-  kibana
 - Window into elastic stack. Enables **visual exploration** and **real-time analysis** of data in Elasticsearch
-  logstash
 - Central dataflow engine for **gathering**, **enriching**, and unifying **input data** from **various sources** (beats, files, REST API) regardless of format or schema and sends it to your favorite stash (elasticsearch, files, email, REST API endpoint, tcp socket, mongodb, [see more](#))
-  beats
 - Forward host-based metrics and any data to Elasticsearch or Logstash
-  x-pack
 - A single extension for **Security** for Elastic Stack, **Alerting** and **notifications** for the Elastic Stack, **Monitoring** for the Elastic Stack, Real-time **Graph Analytics** to enable new use cases for Elastic Stack
-  cloud
 - **Hosted** Elasticsearch & Kibana on **AWS** and **GCP**

Elastic Stack in action

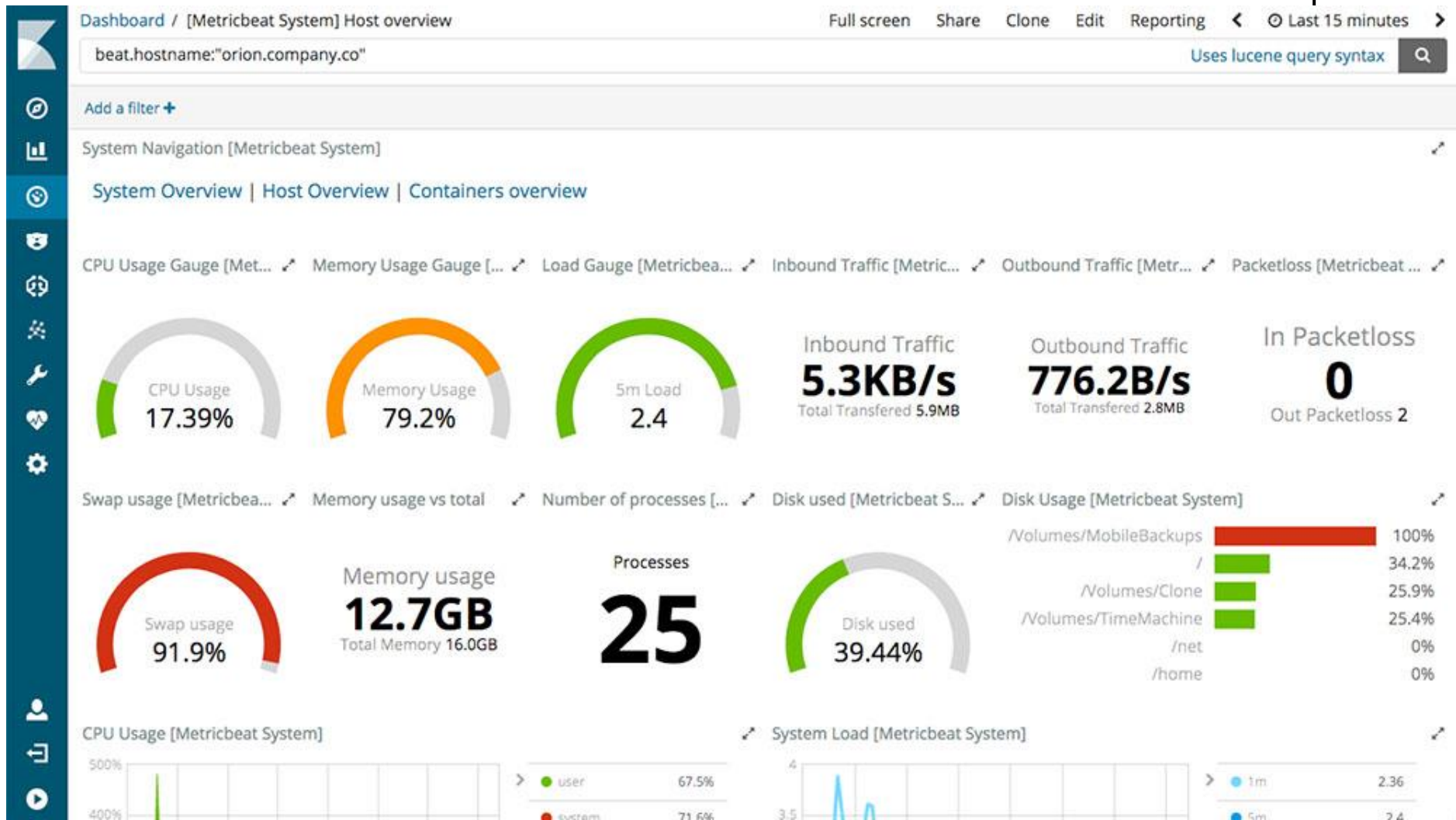


Kibana & Packetbeat in action



Packetbeat is a lightweight network packet analyzer that sends data from your hosts and containers to Logstash or Elasticsearch

Kibana & Metricbeat in action



Collect metrics from your systems and services. From CPU to memory, Redis to NGINX, and much more, Metricbeat is a lightweight way to send system and service statistics.

Kibana & Filebeat in action

A screenshot of the Kibana Logs interface. The top bar shows 'Logs' and a search bar with the placeholder text 'Search for log entries... (e.g. host.name:host-1)'. Below the search bar, there are tabs for 'Default' and 'Customize', and a date/time filter set to '03/26/2019 11:10:45 AM'. The main area displays a list of log entries, each with a timestamp, host name, and log message. The log messages are Ruby stack traces for a Rack application error. The right side of the interface shows a vertical timeline with a blue bar indicating the time range of the logs, from approximately 03 AM to 09 AM on Tuesday, March 26, 2019.

90

Timestamp	Host	Log Message	Time
2019-03-26 11:10:45.445	web.1	2019-03-26 15:10:45 +0000: Rack app error handling request { GET /api/products/top }	Tue 26
2019-03-26 11:10:45.445	web.1	#<HTTP::ConnectionError: failed to connect: getaddrinfo: Name or service not known>	03 AM
2019-03-26 11:10:45.445	web.1	/usr/local/bundle/gems/http-4.0.0/lib/http/timeout/null.rb:21:in `initialize`	03 AM
2019-03-26 11:10:45.445	web.1	/usr/local/bundle/gems/http-4.0.0/lib/http/timeout/null.rb:21:in `open`	03 AM
2019-03-26 11:10:45.445	web.1	/usr/local/bundle/gems/http-4.0.0/lib/http/timeout/null.rb:21:in `connect`	06 AM
2019-03-26 11:10:45.445	web.1	/usr/local/bundle/gems/http-4.0.0/lib/http/connection.rb:43:in `initialize`	06 AM
2019-03-26 11:10:45.445	web.1	/usr/local/bundle/gems/http-4.0.0/lib/http/client.rb:71:in `new`	06 AM
2019-03-26 11:10:45.445	web.1	/usr/local/bundle/gems/http-4.0.0/lib/http/client.rb:71:in `perform`	09 AM
2019-03-26 11:10:45.445	web.1	/usr/local/bundle/gems/elastic-apm-2.0.1/lib/elastic_apm/spies/http.rb:28:in `block in perform`	09 AM
2019-03-26 11:10:45.445	web.1	/usr/local/bundle/gems/elastic-apm-2.0.1/lib/elastic_apm.rb:234:in `with_span`	12 PM
2019-03-26 11:10:45.445	web.1	/usr/local/bundle/gems/elastic-apm-2.0.1/lib/elastic_apm/spies/http.rb:24:in `perform`	12 PM
2019-03-26 11:10:45.445	web.1	/usr/local/bundle/gems/http-4.0.0/lib/http/client.rb:31:in `request`	12 PM
2019-03-26 11:10:45.445	web.1	/usr/local/bundle/gems/http-4.0.0/lib/http/chainable.rb:77:in `request`	03 PM
2019-03-26 11:10:45.446	web.1	/usr/local/bundle/gems/http-4.0.0/lib/http/chainable.rb:20:in `get`	03 PM
2019-03-26 11:10:45.446	web.1	/app/lib/opbeans_shuffle.rb:23:in `block in call`	06 PM
2019-03-26 11:10:45.446	web.1	/usr/local/lib/ruby/2.5.0/timeout.rb:93:in `block in timeout`	06 PM
2019-03-26 11:10:45.446	web.1	/usr/local/lib/ruby/2.5.0/timeout.rb:33:in `block in catch`	09 PM
2019-03-26 11:10:45.446	web.1	/usr/local/lib/ruby/2.5.0/timeout.rb:33:in `catch`	09 PM
2019-03-26 11:10:45.446	web.1	/usr/local/lib/ruby/2.5.0/timeout.rb:33:in `catch`	09 PM
2019-03-26 11:10:45.446	web.1	/usr/local/lib/ruby/2.5.0/timeout.rb:108:in `timeout`	09 PM
2019-03-26 11:10:45.446	web.1	/app/lib/opbeans_shuffle.rb:22:in `call`	09 PM
2019-03-26 11:10:45.446	web.1	/usr/local/bundle/gems/elastic-apm-2.0.1/lib/elastic_apm/middleware.rb:22:in `call`	09 PM
2019-03-26 11:10:45.446	web.1	/usr/local/bundle/gems/railties-5.2.1/lib/rails/engine.rb:524:in `call`	09 PM
2019-03-26 11:10:45.446	web.1	/usr/local/bundle/gems/puma-3.12.0/lib/puma/configuration.rb:225:in `call`	09 PM
2019-03-26 11:10:45.446	web.1	/usr/local/bundle/gems/puma-3.12.0/lib/puma/server.rb:658:in `handle_request`	09 PM

Filebeat offers a lightweight way to forward and centralize logs and files.

Kibana & Filebeat in action



Dashboard / [Logs] Web Traffic

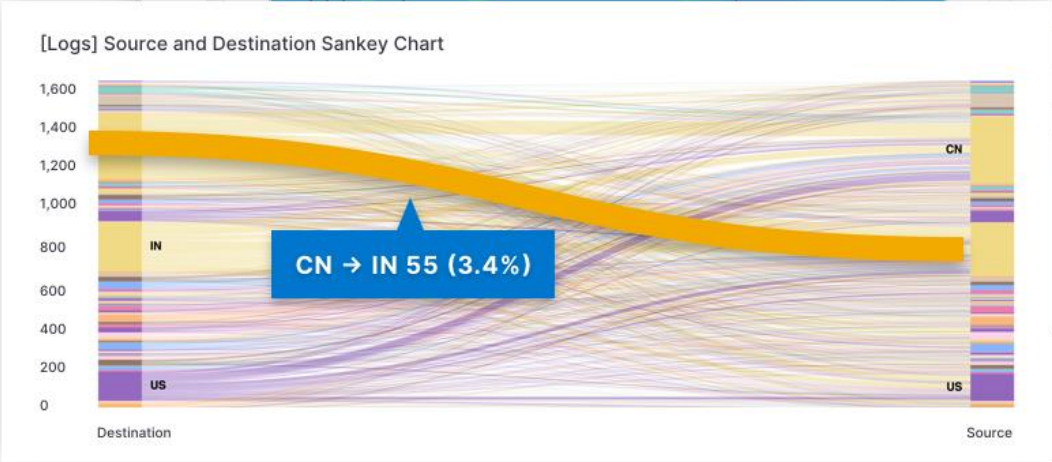
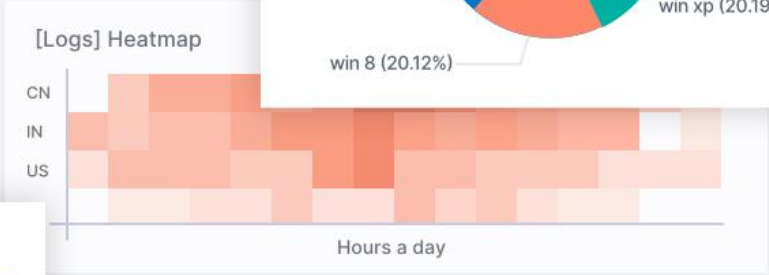
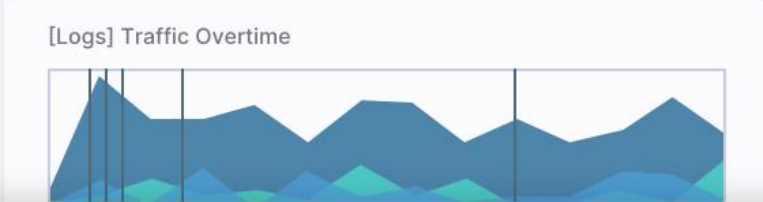
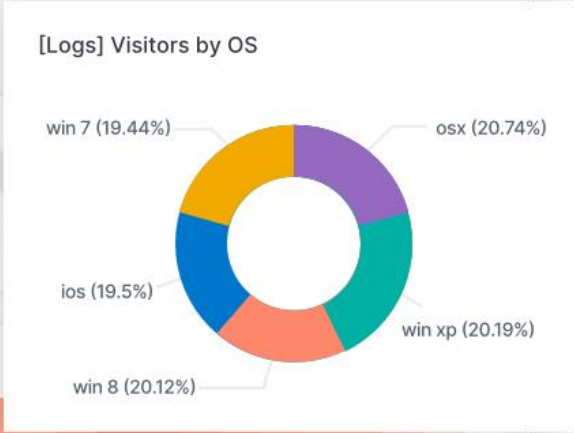
Full screen Share Clone Edit

Filters Search KQL

Last 7 days Show dates Refresh

Source Country OS Bytes

Select... Select... 0 19956



De facto search solution



- **Github** uses Elasticsearch to index and query over 8 million repositories as well as indexing critical event data
- **Wikipedia** uses Elasticsearch to provide **full-text search** with highlighted search snippets, and *search-as-you-type* and *did-you-mean* suggestions
- **Stack Overflow** combines **full-text search** with geolocation queries and uses *more-like-this* to find related questions and answers
- **Vimeo** uses Elasticsearch to make navigation, exploration, and discovery of content as easy and awesome as possible

DB-Engines Ranking of Search Engines



include secondary database models

21 systems in ranking, August 2020

Rank			DBMS	Database Model	Score		
Aug 2020	Jul 2020	Aug 2019			Aug 2020	Jul 2020	Aug 2019
1.	1.	1.	Elasticsearch +	Search engine, Multi-model i	152.32	+0.73	+3.23
2.	2.	2.	Splunk	Search engine	89.91	+1.64	+4.03
3.	3.	3.	Solr	Search engine	51.69	+0.05	-7.43
4.	4.	4.	MarkLogic +	Multi-model i	12.22	+0.55	-2.25
5.	5.	↑ 6.	Microsoft Azure Search	Search engine	6.70	+0.15	-0.14
6.	6.	↓ 5.	Sphinx	Search engine	6.45	-0.09	-0.49
7.	↑ 8.	↑ 8.	Algolia	Search engine	6.14	+0.75	+1.29
8.	↓ 7.	↓ 7.	ArangoDB +	Multi-model i	5.73	-0.11	+0.61
9.	↑ 10.	↑ 10.	Virtuoso +	Multi-model i	2.65	+0.21	-0.41
10.	↓ 9.	↓ 9.	Amazon CloudSearch	Search engine	2.54	-0.13	-0.75
11.	11.	↑ 13.	Xapian	Search engine	0.82	+0.03	0.00
12.	12.	12.	CrateDB +	Multi-model i	0.79	+0.05	-0.12
13.	13.		Alibaba Cloud Log Service +	Search engine	0.33	+0.03	
14.	14.	14.	SearchBlox	Search engine	0.32	+0.02	+0.05
15.	↑ 16.		Weaviate	Search engine, Multi-model i	0.07	+0.01	
16.	↓ 15.	↓ 15.	Manticore Search	Search engine	0.06	-0.01	-0.01
17.	17.	↑ 18.	Exorbyte	Search engine	0.03	-0.02	-0.01
18.	18.	↓ 17.	searchxml	Multi-model i	0.03	+0.00	-0.01
19.	19.	19.	FinchDB	Multi-model i	0.02	-0.01	0.00
20.	20.	20.	Indica	Search engine	0.00	±0.00	±0.00
20.	20.		Rizhiyi	Search engine, Multi-model i	0.00	±0.00	


Elasticsearch Characteristics



- Data sources
 - Elasticsearch accepts data from many different sources such as ActiveMQ, AWS SQS, DynamoDB (Amazon NoSQL), FileSystem, Git, JDBC, JMS, Kafka, LDAP, MongoDB, neo4j, RabbitMQ, Redis, Solr, Twitter, etc.
- Scalable and Distributed Operation
 - ElasticSearch is ***designed for the cloud***
 - simple to scale / attracts use cases where large clusters required
 - Elasticsearch has built-in cluster coordination subsystem
 - every search must be routed to all the right nodes to ensure that its results are accurate
 - every replica must be updated when you index or delete some documents.
 - every client request must be forwarded from the node that receives it to the nodes that can handle it.

Elasticsearch Characteristics



- Searching  (e.g. aggregations)
 - Elasticsearch is often used for analytical querying, filtering, and grouping
 - other than indexing text very well, can also index numbers, dates, geographical coordinates, and almost any other datatype
 - Elasticsearch is always trying to make queries more efficient (through methods including the lowering of memory footprint and CPU usage) and improve performance at both the Lucene and Elasticsearch levels.
 - **Elasticsearch** is a better choice for applications that require **not only text search but also complex timeseries search and aggregations** (similar to GROUP BY)

Elasticsearch Characteristics



- Elasticsearch is very popular among newer developers due to its ease of use
 - Elasticsearch is a very good option for cloud and distributed environments that need good scalability and performance
-

Drawbacks



- Fast for fetching the first 100s-1000s of docs. When querying ~ 10000 results it gets relatively slow
 - Used for simple analytics using ES aggregations feature. Run queries such as: `“Filter all documents from 2016-04-23, and return the sum of the field ‘pages’ from all those documents”`
Works fast because filter and aggregation only run on indexed data. Anything more complex (mass data manipulation, joins, or window functions for example) will not run so well.
-

Building blocks



Term	Description
Cluster	A group of nodes that holds all data
Node	A single machine (server) that holds some data and participates on the cluster's indexing and querying
Index	A collection of documents that have similar characteristics. An index can be divided into multiple pieces called shards .
Document	Basic unit of information that can be indexed. Expressed in JSON. Contains fields in key/value pair(s)
Shard	Primary shard and replica(s) may exist. Each Elasticsearch shard is a Lucene index with an upper limit of docs. Sharding is important for 2 reasons: <ul style="list-style-type: none">• Horizontal splitting/scaling of content volume• Allows to distribute and parallelize operations across shards (potentially on multiple nodes) thus increasing performance/throughput

Shards



- **Number of primary shards** and replica shards can be **defined** per index at the time **index is created**
 - After the **index is created**, you may change the number of replica shards dynamically anytime but **you cannot change the number of primary shards without re-indexing data**
 - Ideal number of shards should be based on the amount of data in an index.
 - optimal shard should hold 30-50GB of data.
 - For example, if you expect to accumulate around 300GB of application logs in a day, having around 10 shards in that index would be reasonable.
-

Building blocks



Cluster

Node 1

Index 1: shard1_primary

Type

Document1

Field1

Field2

Field3

Document2

Field1

Field2

Field3

Node 2

Index 1: shard1_replica

Type

Document1

Field1

Field2

Field3

Document2

Field1

Field2

Field3

Mapping



- Mapping is the process of defining **how** a **document**, and the **fields** it contains, are **stored** and **indexed**
 - For instance, use mappings to define:
 - which string fields should be treated as full text fields.
 - which fields contain numbers, dates, or geolocations.
 - the format of date values.
 - custom rules to control the mapping for dynamically added fields.
-

Mapping



- Mapping definition has:
 - **Metadata fields**
 - customize how a document's associated metadata is treated
 - examples: document's `_index`, `_id`, and `_source` fields.
 - **Fields**
 - A mapping contains a list of fields or properties pertinent to the document. Each field has its own data type.

Mapping



- Index mapping

```
{
  "newsgroups": {
    "mappings": {
      "properties": {
        "full_name": {
          "type": "text"
        },
        "department": {
          "type": "text"
        }
      }
    }
  }
}
```

Example: A mapping for `newsgroups` index, when the type of the inserted document is `_doc`

A mapping type contains a list of fields or properties pertinent to the document. Each field has a data type such as text, keyword, date, long, double, boolean or ip.

Document to be indexed

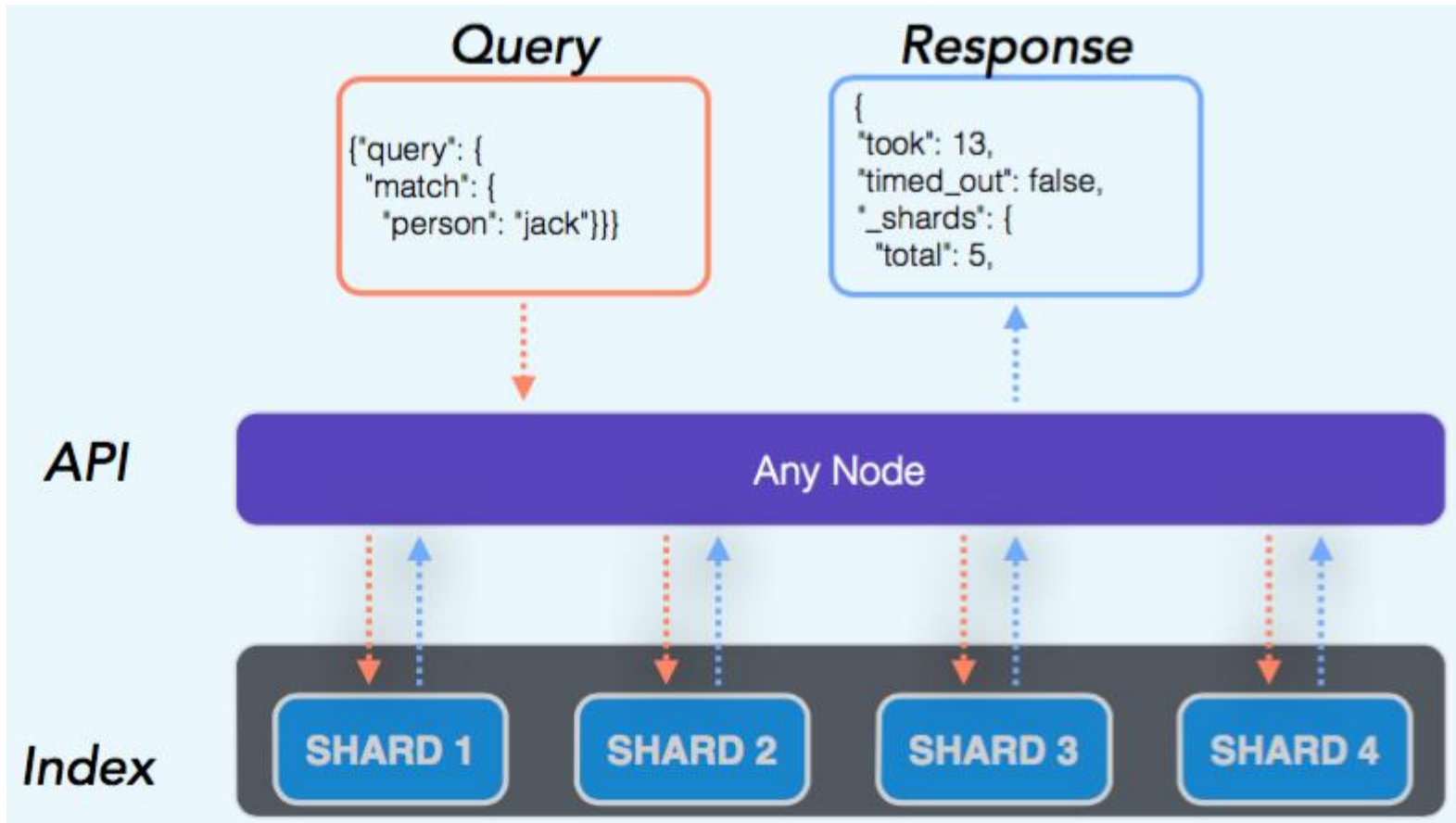
```
{ "_index": "newsgroups", "full_name": "john smith", "department": "computer science" }
```

Dynamic mappings



- Mappings can be explicitly created when creating an index (e.g. via REST API call)
- To index a doc, you don't have to first create an index, define a mapping type, and define your fields
- Every time a document contains new mapping type and new fields, those end up in index's mappings
 - [Lucene StandardAnalyzer](#) for automatic type guessing
- However, defining too many fields in an index is a condition that can lead to a **mapping explosion**, which can cause out of memory errors and difficult situations to recover from
 - Use settings to limit the number of field mappings

Query and response



Hands on: ElasticSearch



- Elasticsearch v7.8.0 installed on VM
 - Kibana installed on VM
 - Logstash installed on VM
-

Hands on: Elasticsearch



- Start Elasticsearch as a service
 - `sudo service elasticsearch start`
- Check if Elasticsearch is working:
 - <http://localhost:9200>

The screenshot shows a Mozilla Firefox browser window with the address bar set to `localhost:9200/`. The page content is displayed in JSON format, showing the following details:

```
{
  "name": "lab-0",
  "cluster_name": "elasticsearch",
  "cluster_uuid": "TF2QgE3MTZ-VA9bNK0ia7Q",
  "version": {
    "number": "7.8.0",
    "build_flavor": "default",
    "build_type": "deb",
    "build_hash": "757314695644ea9a1dc2fec26d1a43856725e65",
    "build_date": "2020-06-14T19:35:50.234439Z",
    "build_snapshot": false,
    "lucene_version": "8.5.1",
    "minimum_wire_compatibility_version": "6.8.0",
    "minimum_index_compatibility_version": "6.0.0-beta1"
  },
  "tagline": "You Know, for Search"
}
```

Hands on: MetricBeat & Kibana



- Metricbeat helps you monitor your servers and the services they host by collecting metrics from the operating system and services.
 - Steps to be followed to use MetricBeat:
 - install Metricbeat on each system you want to monitor
 - specify the metrics you want to collect
 - send the metrics to Elasticsearch
 - visualize the metrics data in Kibana
-

Hands on: MetricBeat & Kibana



- Install Metricbeat on VM
 - Follow steps 1-4 to install Kibana using APT repositories
<https://www.elastic.co/guide/en/beats/metricbeat/current/setup-repositories.html>
 - Start Kibana as a service
 - `sudo service kibana start`
 - Set up the Kibana dashboards for Metricbeat
 - `sudo metricbeat setup --dashboards`
 - Kibana must be running and reachable
 - See list of available modules that collect metrics
 - `sudo metricbeat modules list`
-

Hands on: MetricBeat & Kibana



- You can enable one or more modules:
 - `sudo metricbeat modules enable apache mysql`
 - If you accept the default configuration without enabling additional modules, Metricbeat collects system metrics only.

 - Start Metricbeat as a service
 - `sudo service metricbeat start`
-

Kibana – Discover page



The navigation sidebar of Kibana, showing various sections and their status. A red arrow points to the 'Discover' option.

- Home
- Recently viewed
 - [Metricbeat System] Overview ECS
- Kibana
 - Discover
 - Dashboard
 - Canvas
 - Maps
 - Machine Learning
 - Visualize
- Observability
 - Logs
 - Metrics
 - APM
 - Uptime

The Kibana Discover page interface. The search bar contains 'metricbeat-*' and the time range is set to 'Last 15 minutes'. A bar chart shows the count of hits over time, with a peak around 23:07:00. Below the chart, two log entries are displayed.

Discover - Elastic - Mozilla Firefox

localhost:5601/app/kibana#/discover?_g=(filters:!(),refreshInterval:(pause:!t,value: ...)

Search: metricbeat-* KQL Last 15 minutes Show dates Refresh

metricbeat-* 267 hits

Nov 4, 2020 @ 22:53:48.488 - Nov 4, 2020 @ 23:08:48.488 — Auto

Count

@timestamp per 30 seconds

Time

_source

```
> Nov 4, 2020 @ 23:08:32.593 @timestamp: Nov 4, 2020 @ 23:08:32.593 user.name: ubuntu metricset.name: process
metricset.period: 10,000 service.type: system agent.type: metricbeat agent.version: 7.9.3
agent.hostname: lab-0 agent.ephemeral_id: 303dfde9-fd85-4b02-b8a7-f08d3a04aa5d
agent.id: a39c9303-4aee-4080-b04e-c78eeeb8ad3c agent.name: lab-0 system.process.cmdline: /usr
/lib/firefox/firefox -contentproc -childID 6 -isForBrowser -prefsLen 10600 -prefMapSize 229203

> Nov 4, 2020 @ 23:08:32.593 @timestamp: Nov 4, 2020 @ 23:08:32.593 system.process.state: running
system.process.memory.share: 33.1MB system.process.memory.size: 327.8MB
```

Kibana – Dashboard page



The sidebar navigation menu is located on the left side of the Kibana interface. It includes a search bar at the top, followed by a 'Home' button. Below that is a 'Recently viewed' section showing '[Metricbeat System] Overview ECS'. The main navigation area is divided into two sections: 'Kibana' and 'Observability'. The 'Kibana' section contains links for 'Discover', 'Dashboard', 'Canvas', 'Maps', 'Machine Learning', and 'Visualize'. The 'Observability' section contains links for 'Logs', 'Metrics', 'APM', and 'Uptime'. A red arrow points from the 'Dashboard' link to the main dashboard content area.

The main dashboard area displays the 'Overview ECS' for the 'Metricbeat System'. The page title is '[Metricbeat System] Overview ECS - Elastic - Mozilla Firefox'. The browser address bar shows 'localhost:5601/app/kibana#/dashboard/Metricbeat-system-overview-ecs?_g=(filter...'. The dashboard includes a search bar, a 'KQL' button, and a refresh button. The main content area is titled 'System Navigation [Metricbeat System] ECS' and includes links for 'System Overview', 'Host Overview', and 'Containers overview'. The dashboard features several key metrics: 'Number of hosts [Metricbeat System] ECS' (1), 'CPU Usage Gauge [Metricbeat System] ECS' (60.7%), 'Memory Usage Gauge [Metricbeat System] ECS' (96.9%), 'Disk used [Metricbeat System] ECS' (75.141%), 'Inbound Traffic [Metricbeat System] ECS' (3.9KB/s), and 'Outbound Traffic [Metricbeat System] ECS' (28.8KB/s). Below these metrics are two horizontal bar charts: 'Top Hosts By CPU (Realtime) [Metricbeat System] ECS' and 'Top Hosts By Memory (Realtime) [Metricbeat System] ECS', both showing 99.6% and 96.9% respectively for host 'lab-0'.

Metric	Value
Number of hosts [Metricbeat System] ECS	1
CPU Usage Gauge [Metricbeat System] ECS	60.7%
Memory Usage Gauge [Metricbeat System] ECS	96.9%
Disk used [Metricbeat System] ECS	75.141%
Inbound Traffic [Metricbeat System] ECS	3.9KB/s
Outbound Traffic [Metricbeat System] ECS	28.8KB/s

Host	CPU Usage (%)	Memory Usage (%)
lab-0	99.6%	96.9%

Experimenting with Logstash



- Logstash activation as a service
 - sudo service logstash start
- Alternative way to start logstash: using binaries
 - cd /usr/share/logstash/bin
 - sudo ./logstash -e 'input { stdin {} } output { stdout {} }'
 - -e flag enables specifying configuration directly from command line
 - Pipeline in the example takes input from the standard input, stdin, and moves that input to the standard output, stdout, in a structured format.

After starting Logstash, wait until you see "Successfully started Logstash API endpoint" and enter messages at the command prompt

```
hello world ←
{
    "host" => "lab-0",
    "@timestamp" => 2020-11-04T06:55:37.001Z,
    "message" => "hello world",
    "@version" => "1"
}
this is a test, logstash welcome! ←
{
    "host" => "lab-0",
    "@timestamp" => 2020-11-04T06:56:01.472Z,
    "message" => "this is a test, logstash welcome!",
    "@version" => "1"
}
```

Input at command prompt