Application Delivery Fundamentals

High performance. Delivered.

Parameswari Ettiappan

ELK

Goals

- Why ELK?
- Architecture of ELK
- High level overview of Elastic Search, Logstash, Kibana
- Getting started with Logstash
- Shipping events
- Shipping events without the Logstash agent
- Filtering events
- Structured Application logging.

Goals

- Outputting events
- Scaling Logstash
- Extending Logstash
- Elastic Search Overview
- Installing and running Elastic Search
- Indexing Documents
- Retrieving a Document
- Searching a Document

ELK

Goals

- Analyzers - Tokenizers and Filters
- Character Filters
- Testing Analyzers
- Built-In Analyzers
- Synonym Handling
- CRUD and relationship to documents/indices
- Data Types
- Dynamic Field Mappings
- Index Templates

Goals

- Structured Search
- Full text Search
- Complicated Search
- Phrase Search
- Highlighting our Search
- Multi-field Search
- Proximity Matching
- Partial Matching

ELK

Goals

- Distributed Search Fundamentals
- Query DSL Deep Dive
- Query Advice and Best Practices
- Fundamentals
- Deep dive of each aggregation
- Elastic Search vs RDBMS
- Handling Relationships
- Nested Objects
- Parent-Child Relationship
- API’s
- Designing for Scale

Goals

- Geo Points
- Geo hashes
- Geo Aggregations
- Geo Shapes
- Introduction to Kibana
- Installing Kibana
- Loading Sample Data
- Discovering your Data
- Visualizing your Data
- Working with Dashboard

ELK

Goals

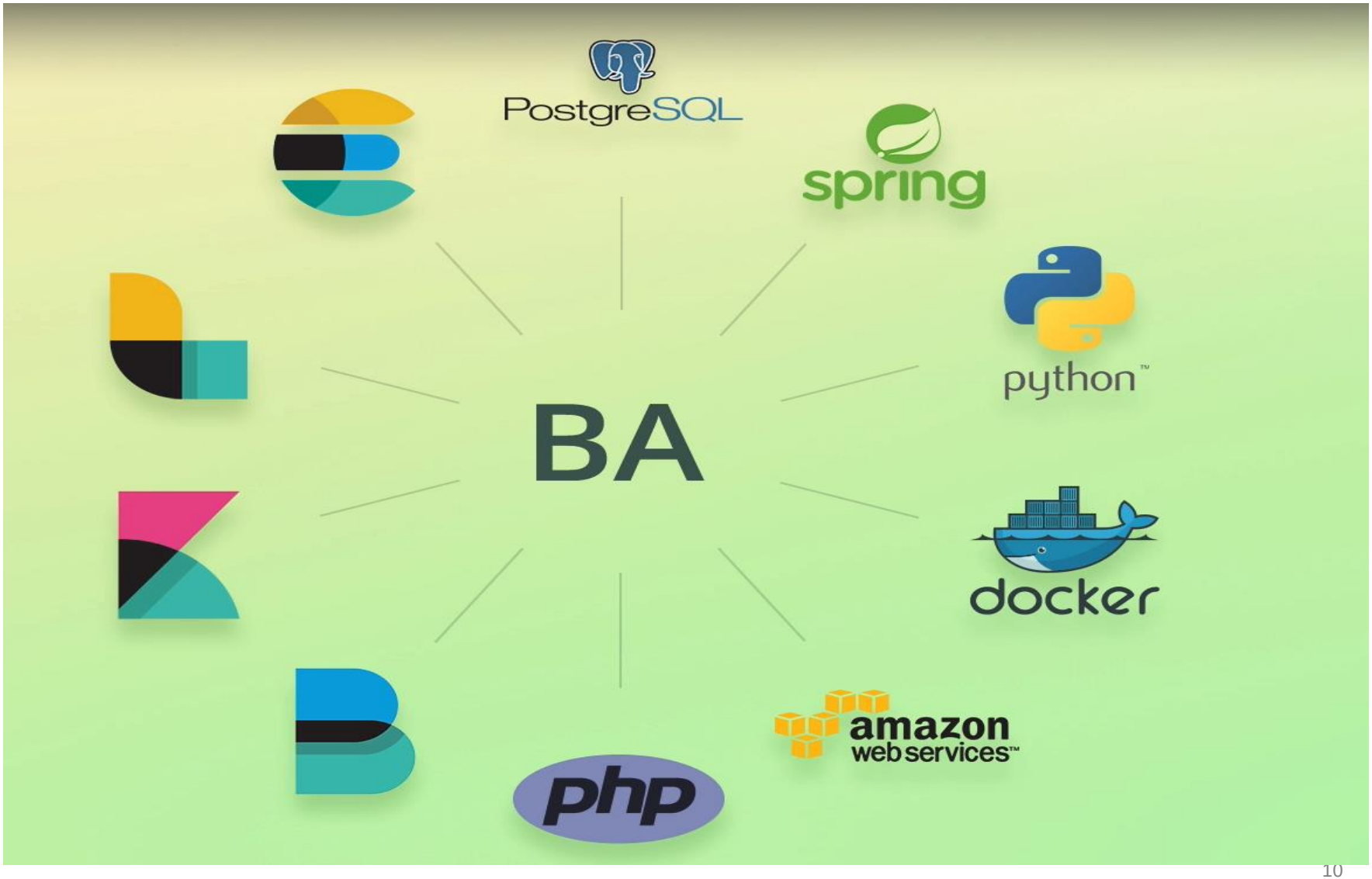
- Setting the Time Filter
- Searching your Data
- Filtering by Field
- Viewing Document Data
- Viewing Document Context
- Viewing Field Statistics
- Data Visualization
- Dashboard
- Analyzing live data with ELK stack

ELK

Goals

- High availability
- Scalability
- Build and configure your first data pipeline with ELK
- Collect, Parse, and Transform Data with Logstash
- Handling Back Pressure
- Deployment Architectures
- Hardware Best Practices
- Security
- Debugging and Monitoring

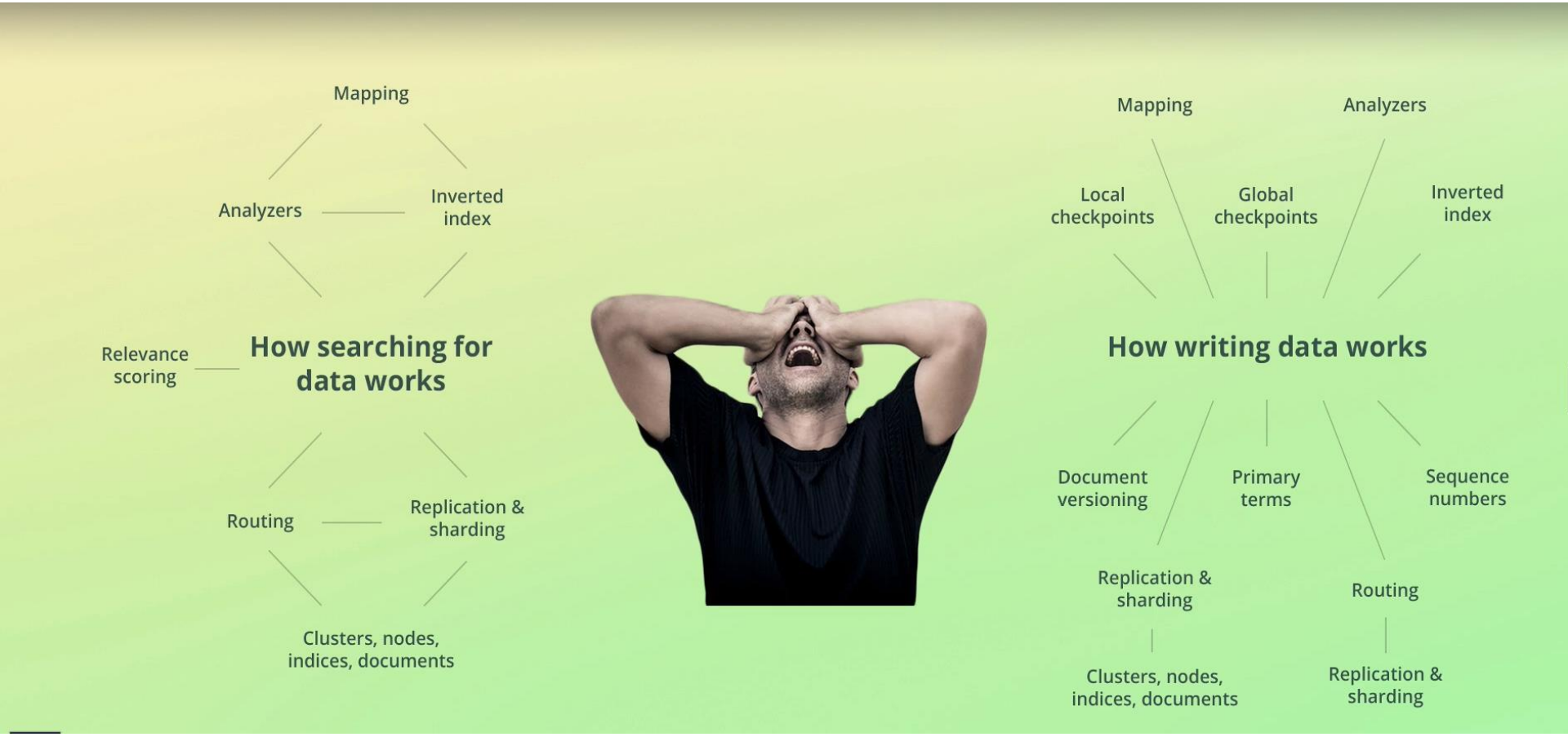
Why I teach ELK



Who Are You

- This course starts at the beginner's level
  - Some experience with Elasticsearch is also fine 🙂
- The course *guides* you through the process of learning Elasticsearch

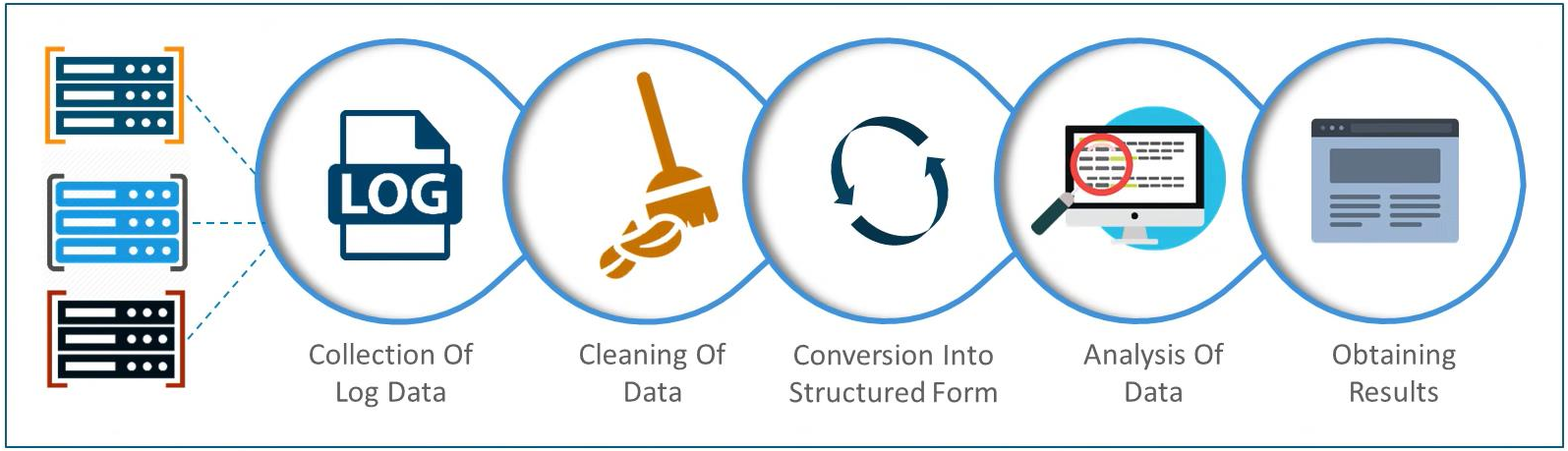
Who Are You


Mapping
Analyzers
Inverted index
Relevance scoring
How searching for data works
Routing
Replication & sharding
Clusters, nodes, indices, documents
Mapping
Analyzers
Local checkpoints
Global checkpoints
Inverted index
How writing data works
Document versioning
Primary terms
Sequence numbers
Replication & sharding
Routing
Clusters, nodes, indices, documents
Replication & sharding

What is Log Analysis?

What is Log Analysis

Log Analysis Is The Process Of Analyzing The Computer/ Machine Generated Data



Need for Log Analysis

Log Analysis

- Almost all kinds of computing devices, systems, and applications emit some kind of log to indicate the state of the system.
- Put simply, a log is just a stream of messages in a time sequence.
- They may be directed to files and stored on a disk or directed to a log collector.
- Raw logs are just data, but when they are processed and analyzed, they provide useful information.
- Log analyzers take as input the mass of data produced by our firewalls, routers, IDS, and applications, and turn that data into actionable intelligence.

Log Analysis

- Whenever a developer or system admin faces an issue with the system, the first instinct is to look at the logs.
- For a long time, we have relied on basic tools like grep, awk, or perl to perform log analysis.
- However, with changing times and cloud-scale applications, the earlier techniques no longer suffice.
- Imagine a system with tens, hundreds, or thousands of hosts.
- There are multiple instances of different applications running on all these hosts.

Log Analysis

- Whenever a developer or system admin faces an issue with the system, the first instinct is to look at the logs.
- For a long time, we have relied on basic tools like grep, awk, or perl to perform log analysis.
- However, with changing times and cloud-scale applications, the earlier techniques no longer suffice.
- Imagine a system with tens, hundreds, or thousands of hosts.
- There are multiple instances of different applications running on all these hosts.

Log Analysis

- In such a world, it is not possible to troubleshoot problems by using earlier-used tools or just looking at one particular host.
- Add to this the fact that the logs may be generated in different time zones, formats, and even in different languages.
- What is required a holistic log generation, parsing, storage, and analysis solution.

Log Analysis

- More and more IT infrastructure is moving to public clouds such as Amazon Web Services and Microsoft Azure, making log analytics platforms more and more critical.
- Performance isolation is not trivial in cloud-based infrastructures.
- There are many factors for this, like load fluctuation on virtual machines, the dynamic number of users, and change in the environment.
- These issues can be monitored only by a next generation log management platform that can scan through different sources like system logs, web server logs, application logs, and ELB and S3 logs on AWS.
- Proper log analysis can help DevOps engineers, system administrators, site reliability engineers, and developers to make better decisions.

Log Analysis

- The following are some common use cases where log analysis is helpful:
  - Issue debugging
  - Performance analysis
  - Security analysis
  - Predictive analysis
  - Internet of things (IoT) and logging

Log Analysis

Issue debugging

- Debugging is one of the most common reasons to enable logging within your application.
- The simplest and most frequent use for a debug log is to grep for a specific error message or event occurrence.
- If a system administrator believes that a program crashed because of a network failure, then he or she will try to find a connection dropped message or a similar message in the server logs to analyze what caused the issue.
- Once the bug or the issue is identified, log analysis solutions help capture application information and snapshots of that time can be easily passed across development teams to analyze it further.

Log Analysis

Performance analysis

- Log analysis helps optimize or debug system performance and give essential inputs around bottlenecks in the system.
- Understanding a system's performance is often about understanding resource usage in the system.
- Logs can help analyze individual resource usage in the system, behavior of multiple threads in the application, potential deadlock conditions, and so on.
- Logs also carry with them timestamp information, which is essential to analyze how the system is behaving over time.
- For instance, a web server log can help know how individual services are performing based on response times, HTTP response codes, and so on.

Log Analysis

Security analysis

- Logs play a vital role in managing the application security for any organization.
- They are particularly helpful to detect security breaches, application misuse, malicious attacks, and so on.
- When users interact with the system, it generates log events, which can help track user behavior, identify suspicious activities, and raise alarms or security incidents for breaches.
- The intrusion detection process involves session reconstruction from the logs itself.
- For example, ssh login events in the system can be used to identify any breaches on the machines.

Log Analysis

Predictive analysis

- Predictive analysis is one of the hot trends of recent times.
- Logs and events data can be used for very accurate predictive analysis.
- Predictive analysis models help in identifying potential customers, resource planning, inventory management and optimization, workload efficiency, and efficient resource scheduling.
- It also helps guide the marketing strategy, user-segment targeting, ad-placement strategy, and so on.

Log Analysis

- **Internet of things and logging**
- When it comes to IoT devices, it is vital that the system is monitored and managed to keep downtime to a minimum and resolve any important bugs or issues swiftly.
- Since these devices should be able to work with little human intervention and may exist on a large geographical scale, log data is expected to play a crucial role in understanding system behavior and reducing downtime.

Challenges in log analysis

- The current log analysis process mostly involves checking logs at multiple servers that are written by different components and systems across your application.
- This has various problems, which makes it a time-consuming and tedious job.

Let's look at some of the common problem scenarios:

- Non-consistent log format
- Decentralized logs
- Expert knowledge requirement

Challenges in log analysis

Non-consistent log format

- Every application and device logs in its own special way, so each format needs its own expert.
- Also, it is difficult to search across because of different formats.
- Let's look at some of the common log formats.
- An interesting thing to observe will be the way different logs represent different timestamp formats, different ways to represent INFO, ERROR, and so on, and the order of these components with logs.
- It's difficult to figure out just by seeing logs what is present at what location.
- This is where tools such as Logstash help.

Challenges in log analysis

- **Tomcat logs**
- A typical tomcat server startup log entry will look like this:
- **May 24, 2015 3:56:26 PM org.apache.catalina.startup.HostConfig deployWAR**
- **INFO: Deployment of web application archive \soft\apache-tomcat-7.0.62\ webapps\sample.war has finished in 253 ms**

Challenges in log analysis

- **Apache access logs – combined log format**
- A typical Apache access log entry will look like this:
- **127.0.0.1 - - [24/May/2015:15:54:59 +0530] "GET /favicon.ico HTTP/1.1" 200 21630**
- **IIS logs**
- A typical IIS log entry will look like this:
- **2012-05-02 17:42:15 172.24.255.255 - 172.20.255.255 80 GET /images/ favicon.ico - 200 Mozilla/4.0+(compatible;MSIE+5.5;+Windows+2000+Server)**

Challenges in log analysis

- **Variety of time formats**
- Not only log formats, but timestamp formats are also different among different types of applications, different types of events generated across multiple devices, and so on. Different types of time formats across different components of your system also make it difficult to correlate events occurring across multiple systems at the same time:
- 142920788
- Oct 12 23:21:45
- [5/May/2015:08:09:10 +0000]
- Tue 01-01-2009 6:00
- 2015-05-30 T 05:45 UTC
- Sat Jul 23 02:16:57 2014
- 07:38, 11 December 2012 (UTC)
-

Challenges in log analysis

- **Decentralized logs**
- Logs are mostly spread across all the applications that may be across different servers and different components.
- The complexity of log analysis increases with multiple components logging at multiple locations.
- For one or two servers' setup, finding out some information from logs involves running cat or tail commands or piping these results to grep command.
- But what if you have 10, 20, or say, 100 servers? These kinds of searches are mostly not scalable for a huge cluster of machines and need a centralized log management and an analysis solution.

Challenges in log analysis

Expert knowledge requirement

- People interested in getting the required business-centric information out of logs.
- If they don't have access to the logs or may not have the technical expertise to figure out the appropriate information in the quickest possible way, which can make analysis slower, and sometimes, impossible too.

Problems with Log Analysis

1. Non-consistent log format
2. Non-consistent time format
3. Decentralized logs
4. Expert knowledge requirement

Problems with Log Analysis

1. Non-consistent log format
2. Non-consistent time format
3. Decentralized logs
4. Expert knowledge requirement

Tomcat Logs

```
ffMay 24, 2015 3:56:26 PM org.apache.catalina.startup.HostConfig  deployWAR
INFO: Deployment  of web application archive \soft\apache-tomcat-7.0.62\webapps\sample.war
has finished in 253 ms
```

Apache Access Logs

```
127.0.0.1 - - [24/May/2015:15:54:59 +0530] "GET /favicon.ico HTTP/1.1"  200 21630
```

IIS Logs

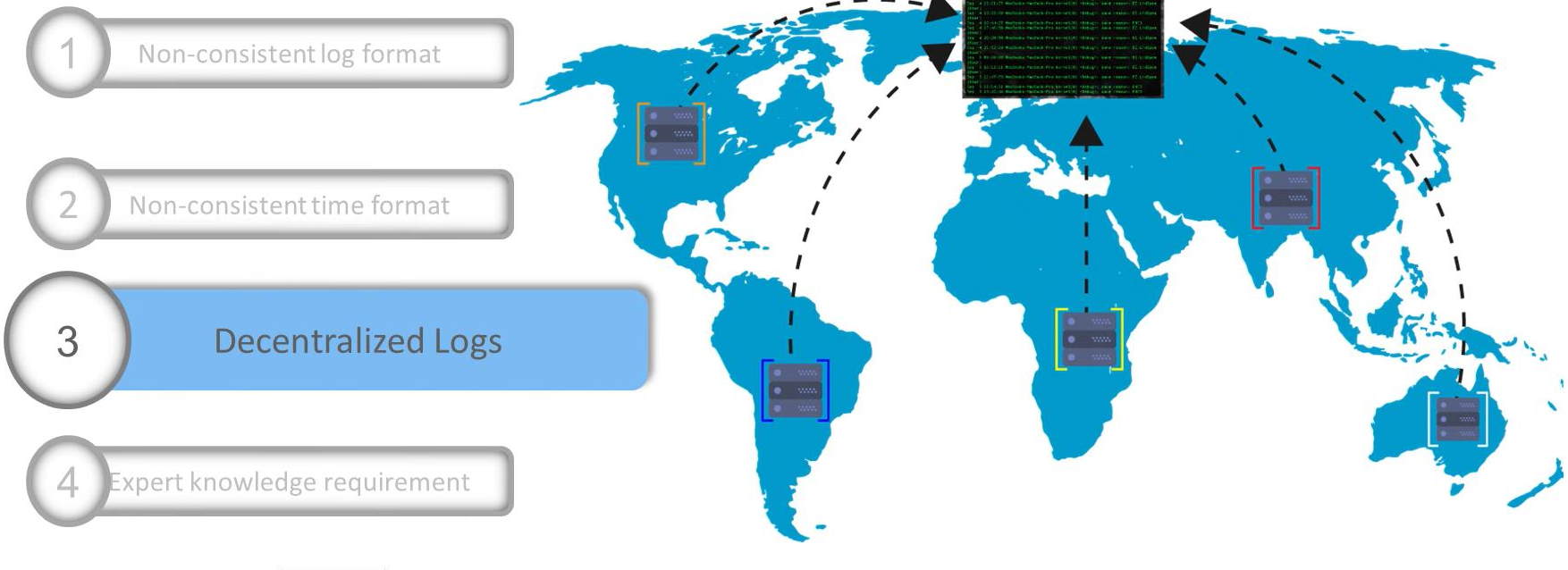
```
2012-05-02 17:42:15 172.24.255.255 - 172.20.255.255 80 GET  /images/favicon.ico - 200
Mozilla/4.0+(compatible;MSIE+5.5;+Windows+2000+Server)
```

Problems with Log Analysis

1. Non-consistent log format
2. **Non-consistent time format**
3. Decentralized logs
4. Expert knowledge requirement

- 142920788
- Oct 12 23:21:45
- [5/May/2015:08:09:10 +0000]
- Tue 01-01-2009 6:00
- 2015-05-30 T 05:45 UTC
- Sat Jul 23 02:16:57 2014
- 07:38, 11 December 2012 (UTC)

Problems with Log Analysis



Problems with Log Analysis

1. Non-consistent log format
2. Non-consistent time format
3. Decentralized logs
4. **Expert Knowledge Requirement**

- Everyone do not have access to the logs
- General people might not have technical expertise to understand the information
- This can slow down the analysis process

Log Management Tools

splunk>

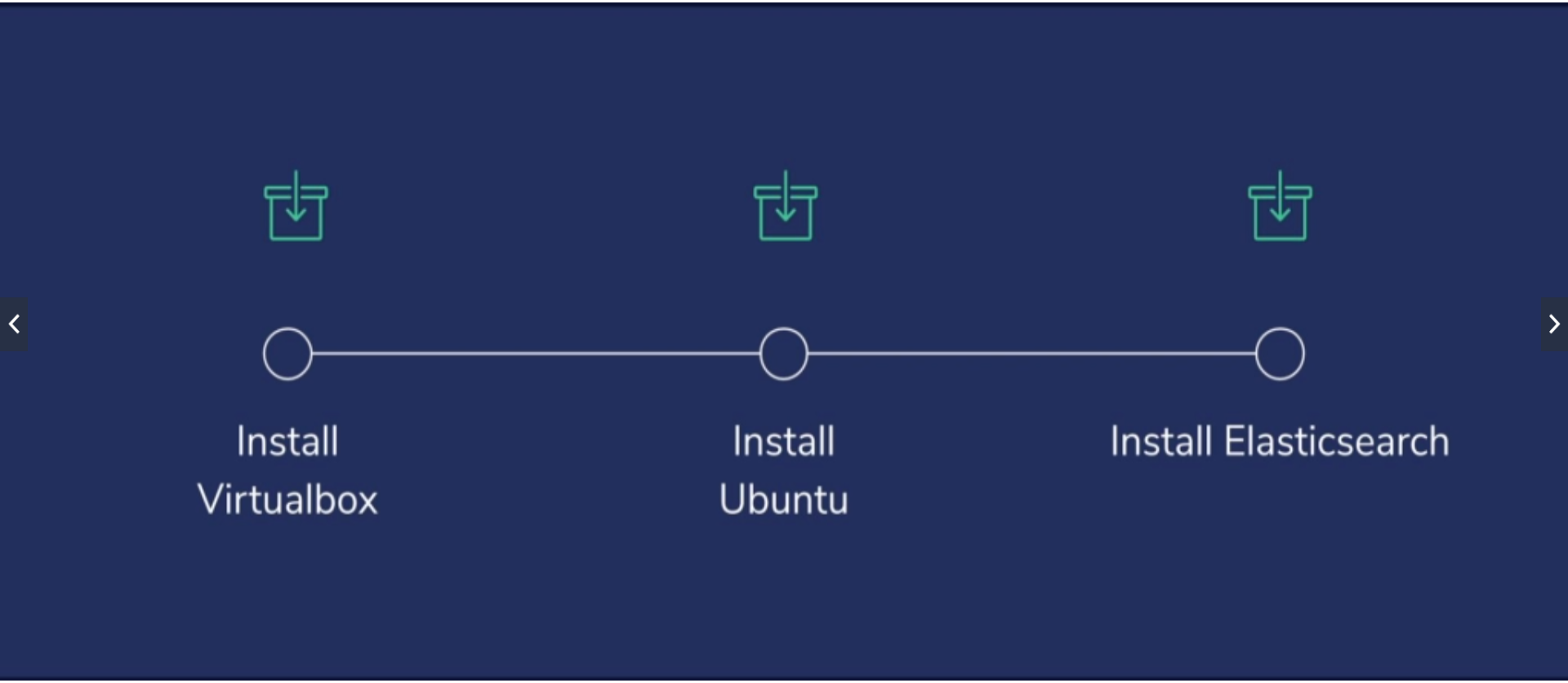
logentries™

graylog

sumologic

LOGGLY

Installing ELK 7



System Requirements

Enable Virtualization

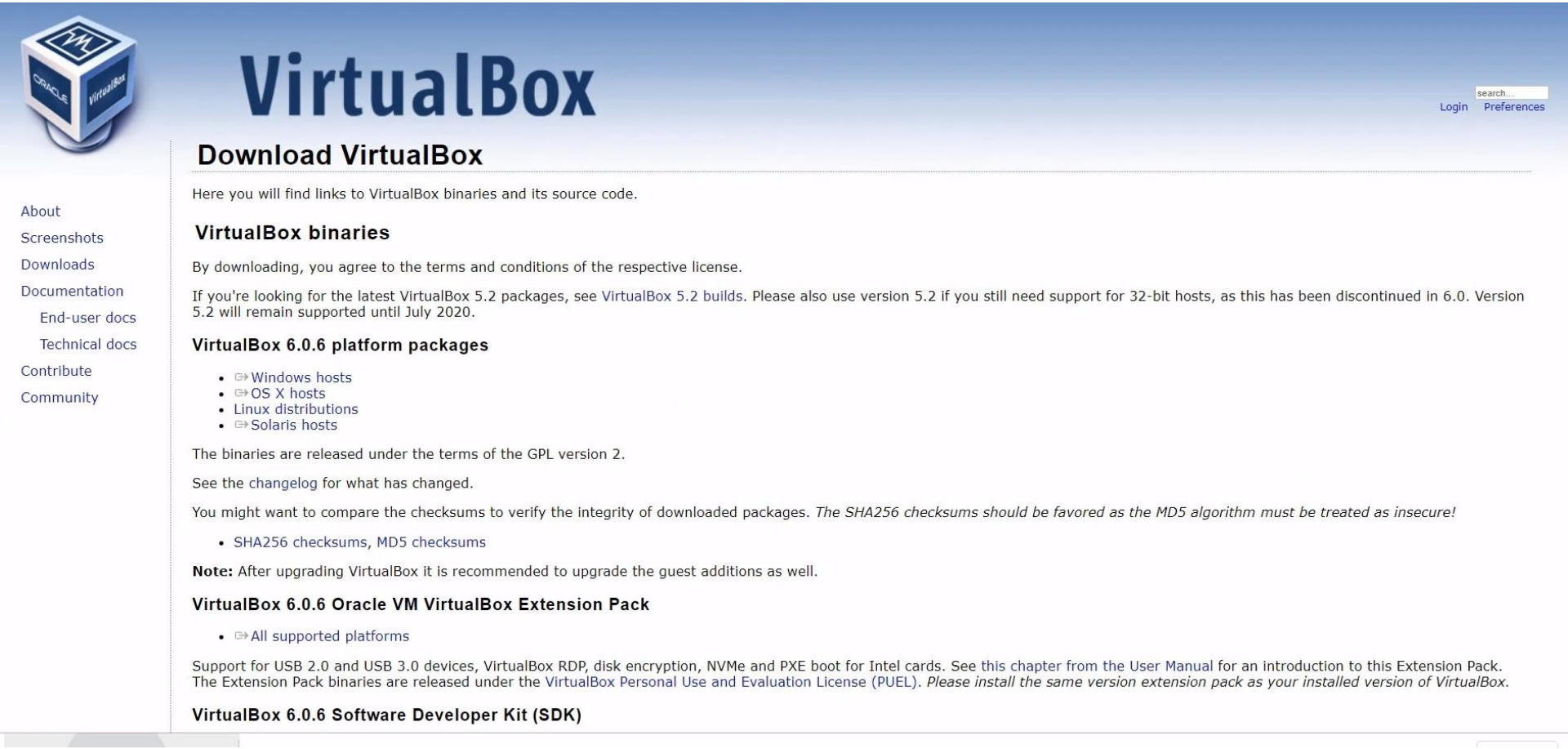
Virtualization must be enabled in your BIOS settings. If you have “Hyper-V” virtualization as an option, turn it off.

Beware Avast

Avast anti-virus is known to conflict with Virtualbox.

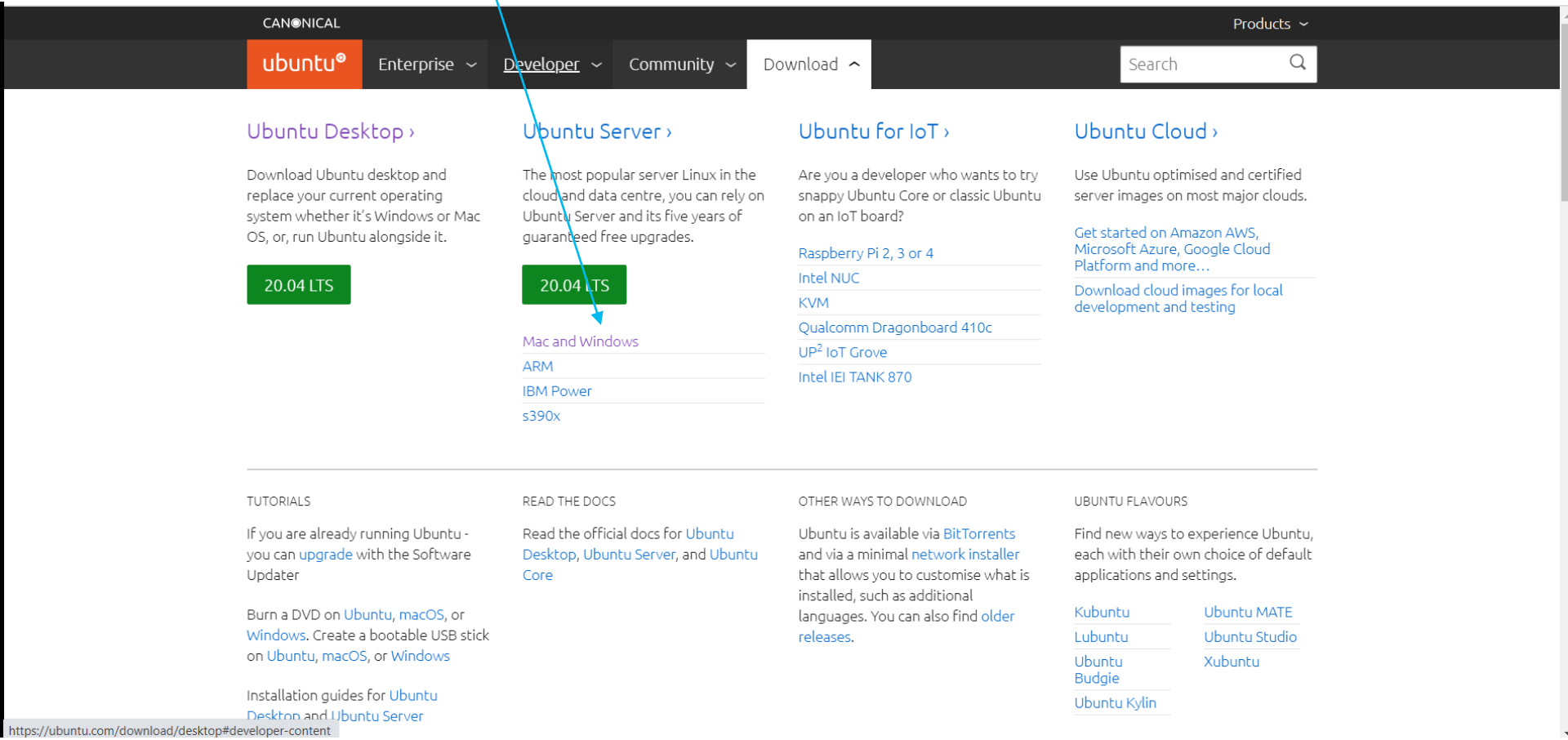
Virtual Box (Mac and Windows)

Virtual Box (Mac and Windows)

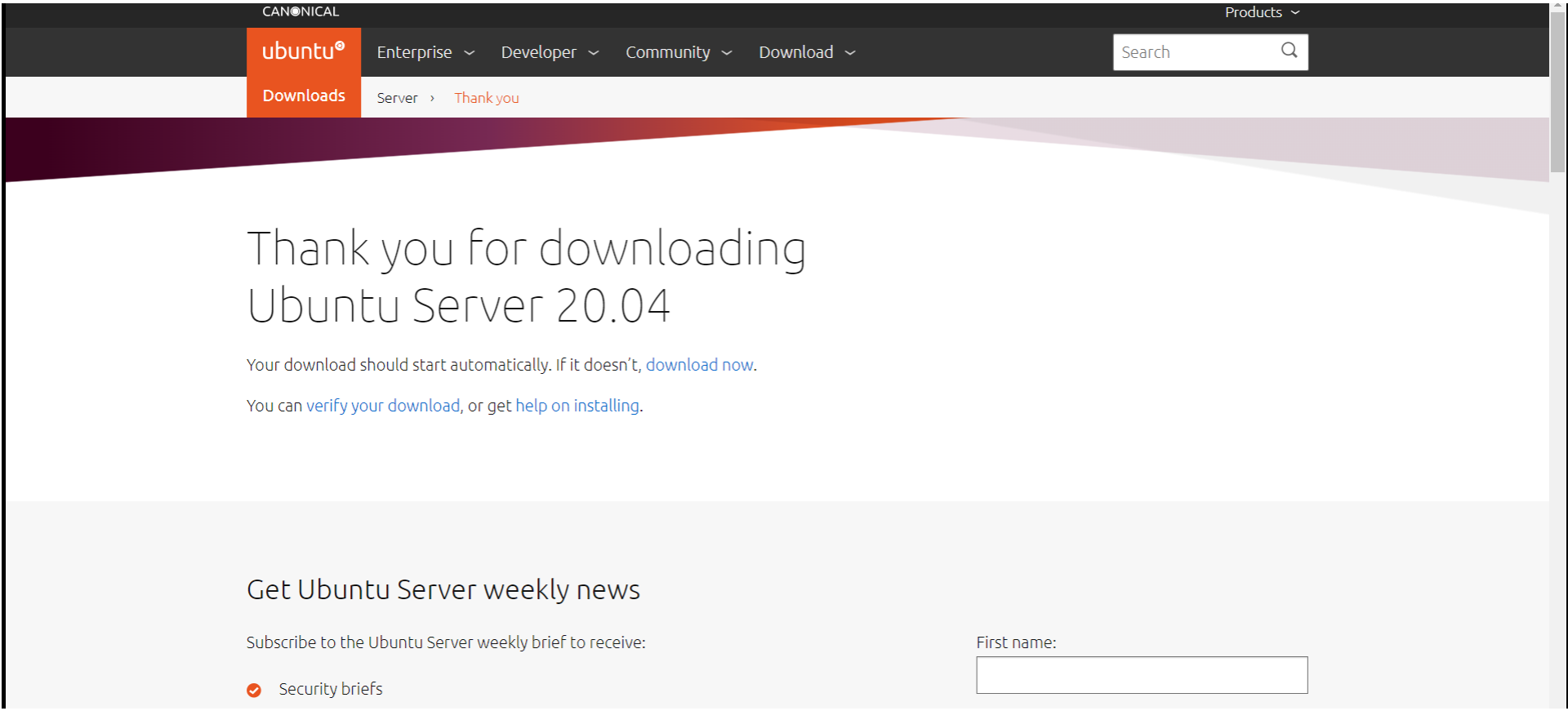


Virtual Box (Mac and Windows)

https://ubuntu.com/download/server/thank-you?version=20.04&architecture=amd64

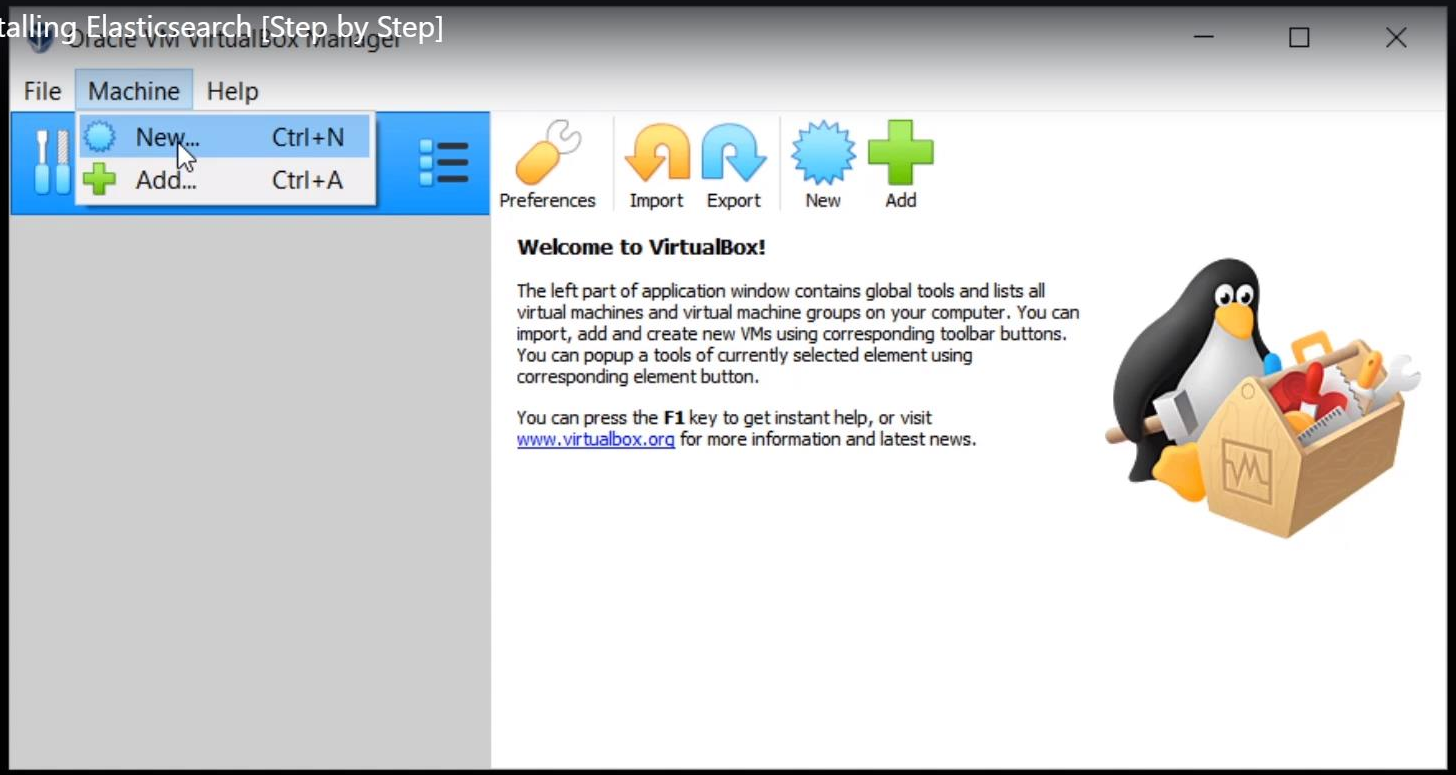


Ubuntu

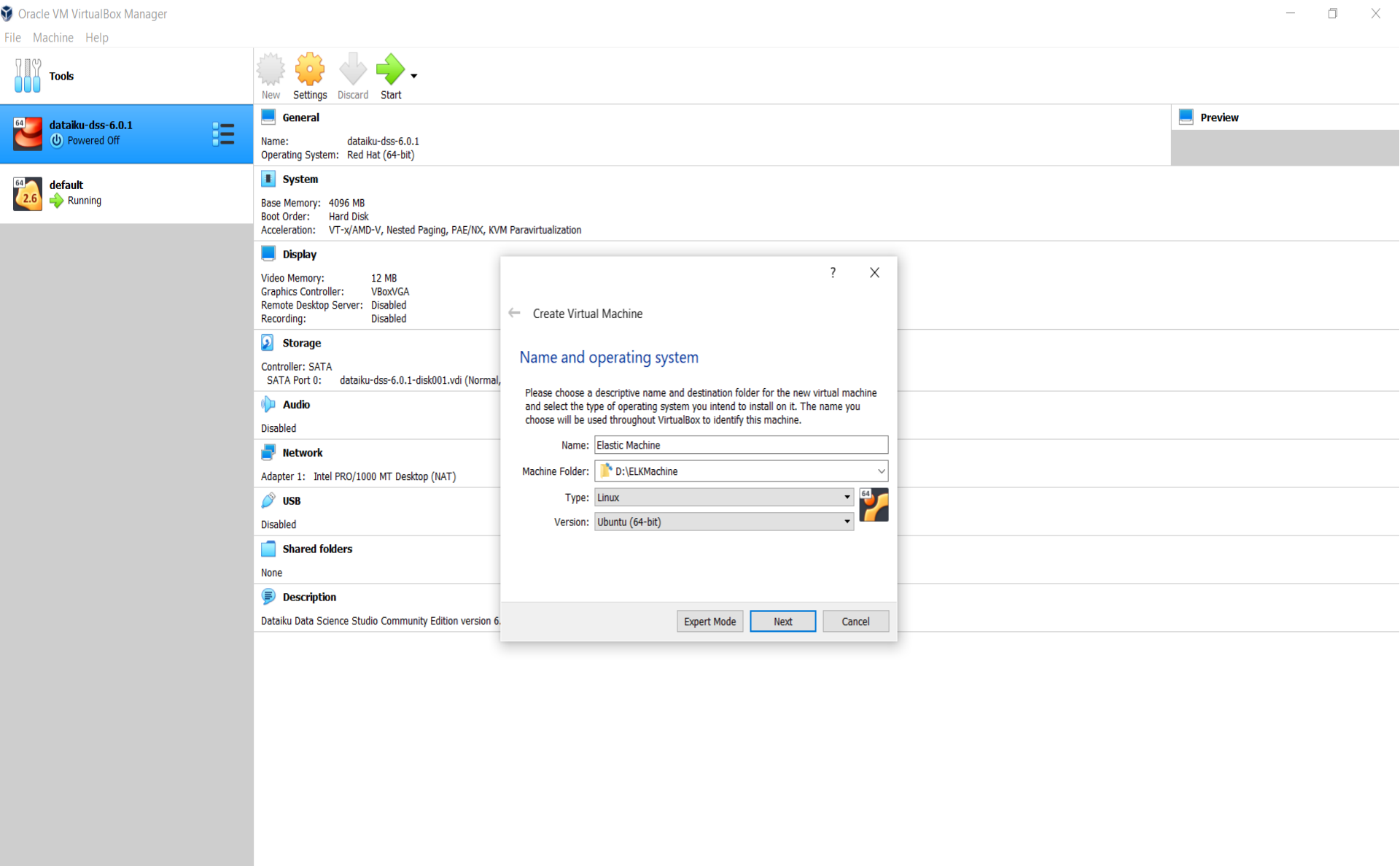

CANONICAL
Products
ubuntu
Enterprise
Developer
Community
Download
Search
Downloads
Server › Thank you
Thank you for downloading Ubuntu Server 20.04
Your download should start automatically. If it doesn't, download now.
You can verify your download, or get help on installing.
Get Ubuntu Server weekly news
Subscribe to the Ubuntu Server weekly brief to receive:
Security briefs
First name:

Ubuntu

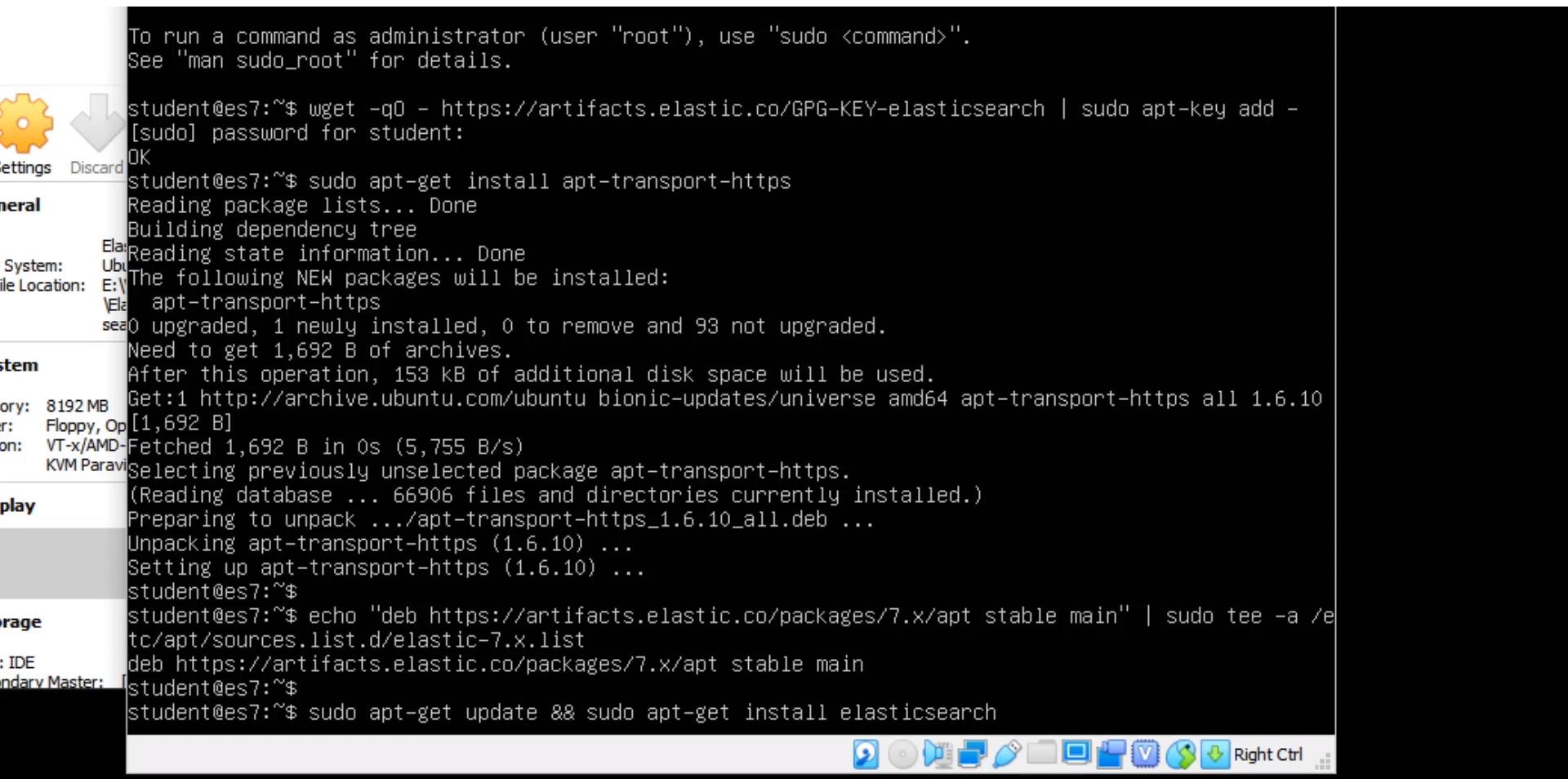
3. Installing Elasticsearch [Step by Step]



Ubuntu


Oracle VM VirtualBox Manager
File Machine Help
Tools
dataiku-dss-6.0.1
Powered Off
default
Running
New Settings Discard Start
General
Name: dataiku-dss-6.0.1
Operating System: Red Hat (64-bit)
Preview
System
Base Memory: 4096 MB
Boot Order: Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX, KVM Paravirtualization
Display
Video Memory: 12 MB
Graphics Controller: VBoxVGA
Remote Desktop Server: Disabled
Recording: Disabled
Storage
Controller: SATA
SATA Port 0: dataiku-dss-6.0.1-disk001.vdi (Normal,
Audio
Disabled
Network
Adapter 1: Intel PRO/1000 MT Desktop (NAT)
USB
Disabled
Shared folders
None
Description
Dataiku Data Science Studio Community Edition version 6.
Create Virtual Machine
Name and operating system
Please choose a descriptive name and destination folder for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.
Name: Elastic Machine
Machine Folder: D:\ELKMachine
Type: Linux
Version: Ubuntu (64-bit)
Expert Mode
Next
Cancel

Ubuntu


To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
student@es7:~$ wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
[sudo] password for student:
OK
student@es7:~$ sudo apt-get install apt-transport-https
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
apt-transport-https
0 upgraded, 1 newly installed, 0 to remove and 93 not upgraded.
Need to get 1,692 B of archives.
After this operation, 153 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 apt-transport-https all 1.6.10
[1,692 B]
Fetched 1,692 B in 0s (5,755 B/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 66906 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_1.6.10_all.deb ...
Unpacking apt-transport-https (1.6.10) ...
Setting up apt-transport-https (1.6.10) ...
student@es7:~$
student@es7:~$ echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a /e
tc/apt/sources.list.d/elastic-7.x.list
deb https://artifacts.elastic.co/packages/7.x/apt stable main
student@es7:~$
student@es7:~$ sudo apt-get update && sudo apt-get install elasticsearch
Right Ctrl

Ubuntu

Curl –XGET http://localhost:9200

```
Microsoft Windows [Version 10.0.18363.836]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Balasubramaniam>curl -XGET http://localhost:9200
{
  "name" : "DESKTOP-55AGI0I",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "vGXcG0gfQtW8Xpch8i7atw",
  "version" : {
    "number" : "7.7.0",
    "build_flavor" : "default",
    "build_type" : "zip",
    "build_hash" : "81a1e9eda8e6183f5237786246f6dced26a10eaf",
    "build_date" : "2020-05-12T02:01:37.602180Z",
    "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"
}

C:\Users\Balasubramaniam>
```

Ubuntu

Wget http://media.sun-soft.com/es1/shakes-mapping.json

Administrator: Command Prompt

```
C:\WINDOWS\system32>wget http://media.sun-soft.com/es1/shakes-mapping.json
--2020-05-30 22:09:31--  http://media.sun-soft.com/es1/shakes-mapping.json
Resolving media.sun-soft.com (media.sun-soft.com)... 15.222.116.2, 15.223.13.12
Connecting to media.sun-soft.com (media.sun-soft.com)|15.222.116.2|:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: http://sun-soft.com/index.php [following]
--2020-05-30 22:09:32--  http://sun-soft.com/index.php
Resolving sun-soft.com (sun-soft.com)... 15.222.116.2, 15.223.13.12
Reusing existing connection to media.sun-soft.com:80.
HTTP request sent, awaiting response... 200 OK
Length: 5652 (5.5K) [text/html]
Saving to: 'shakes-mapping.json.1'

shakes-mapping.json.1             100%[=============================================================>]   5.52K  --.-KB/s    in 0s

2020-05-30 22:09:32 (67.3 MB/s) - 'shakes-mapping.json.1' saved [5652/5652]


C:\WINDOWS\system32>
```

Type here to search

ENG 22:09 30/05/2020

Anatomy of Http Request

METHOD:	The “verb” of the request. GET, POST, PUT, or DELETE
PROTOCOL:	What flavor of HTTP (HTTP/1.1)
HOST:	What web server you want to talk to
URL:	What resource is being requested
BODY:	Extra data needed by the server
HEADERS:	User-agent, content-type, etc.

Anatomy of Http Request

Rest Fancy-speak

Representational State Transfer

Six guiding constraints:

- Client-server architecture
- Statelessness
- Cacheability
- Layered system
- Code on demand (ie, sending Javascript)
- Uniform interface

Why REST

Language and system independent

Curl

The Curl Command

A way to issue HTTP requests from the command line
From code, you'll use whatever library you use for HTTP / REST in the same way.

```
curl –H “Content-Type: application/json” <URL> -d ‘<BODY>’
```

Examples

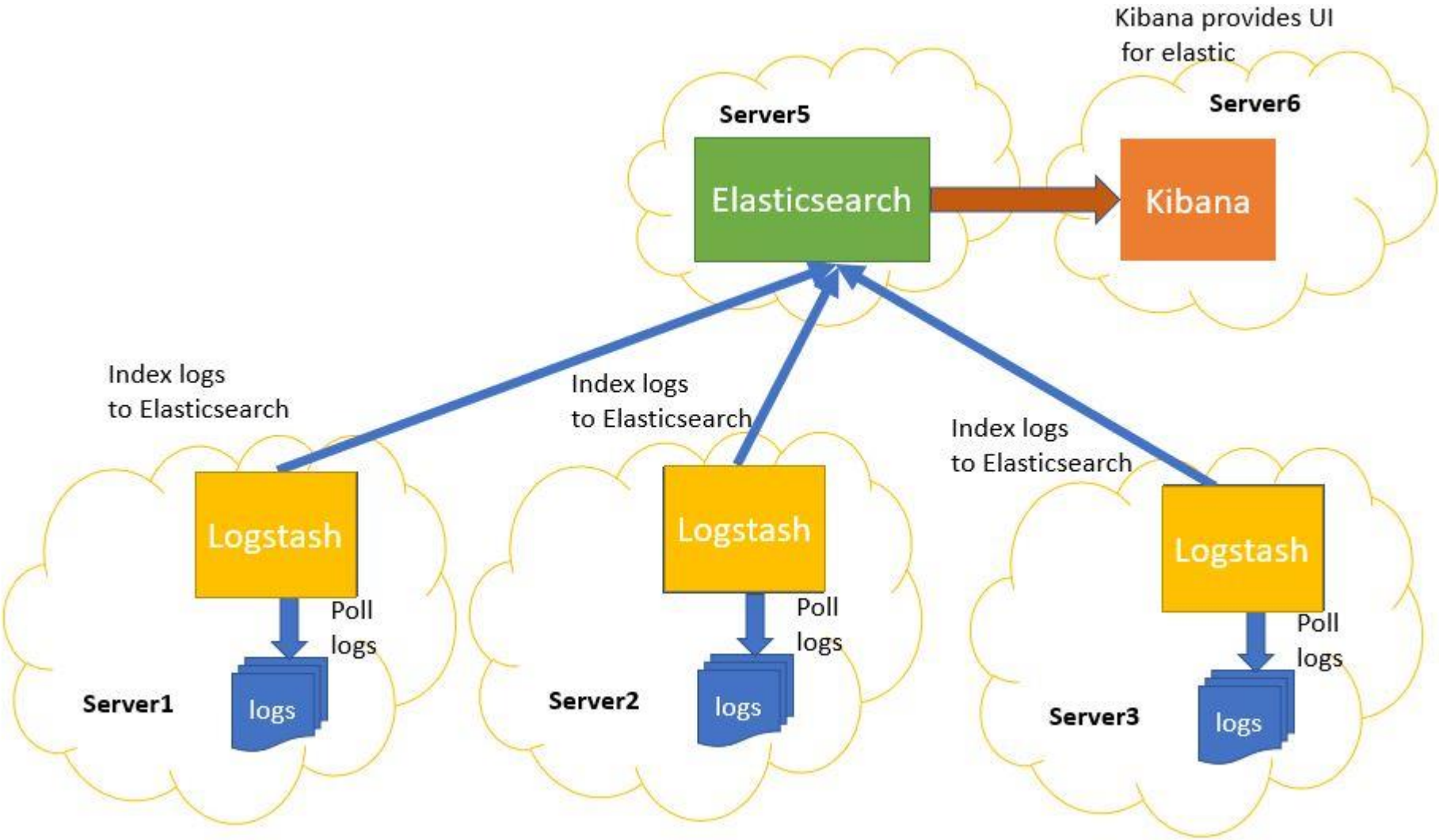
```
curl  -H ‘Content-Type: application/json” -XGET
'127.0.0.1:9200/shakespeare/_search?pretty' -d '
{
    "query" : {
        "match_phrase" : {
            "text_entry" : "to be or not to be"
        }
    }
}
'
```

```
curl -H ‘Content-Type: application/json” -XPUT
127.0.0.1:9200/movies/movie/109487 –d ‘
{
"genre"  : ["IMAX","Sci-Fi"],
"title"  : "Interstellar",
"year"      : 2014
}
```

ELK Stack

- The ELK Stack (Elastic search, Logstash, and Kibana) is made up of open source projects that take data from any source and any format and then search, analyze, and visualize it in real time.
- It offers a next generation log management platform which addresses the issues associated with heterogeneity and scale of logs.
- At the heart of the ELK stack is Elastic search, which is a distributed, open source search and analytics engine

What is ELK


Kibana provides UI
for elastic
Server5
Server6
Elasticsearch
Kibana
Index logs
to Elasticsearch
Index logs
to Elasticsearch
Index logs
to Elasticsearch
Logstash
Logstash
Logstash
Poll
logs
Poll
logs
Poll
logs
Server1
logs
Server2
logs
Server3
logs

Apache Lucene

→ Lucene is a powerful Java search library that lets you easily add search or Information Retrieval (IR) to applications

→ Used by LinkedIn, Twitter, ...and many more

» For more information go to: http://wiki.apache.org/lucene-java/PoweredBy)

→ Scalable and High-performance indexing

→ Powerful, accurate and efficient search algorithms

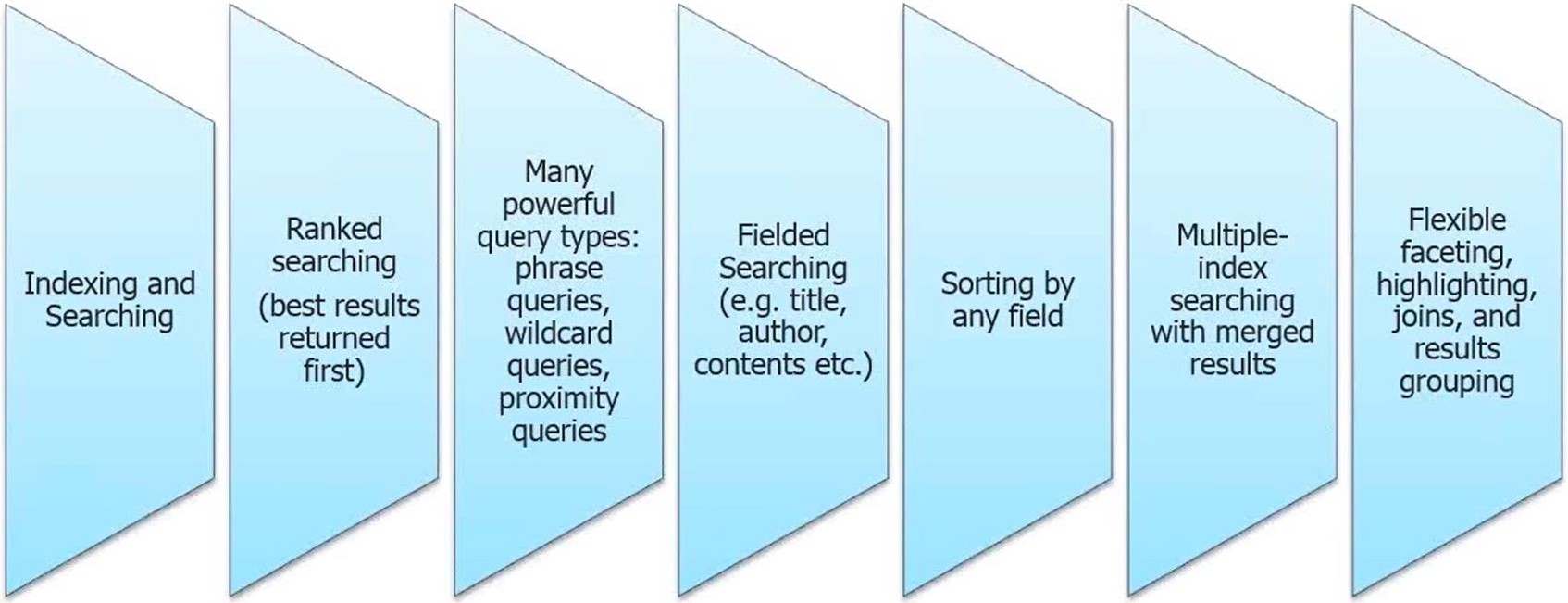
→ Cross-platform solution

» Open source and 100% pure Java

» Implementations in other programming languages available that are index-compatible

Doug Cutting "Creator"

Apache Lucene



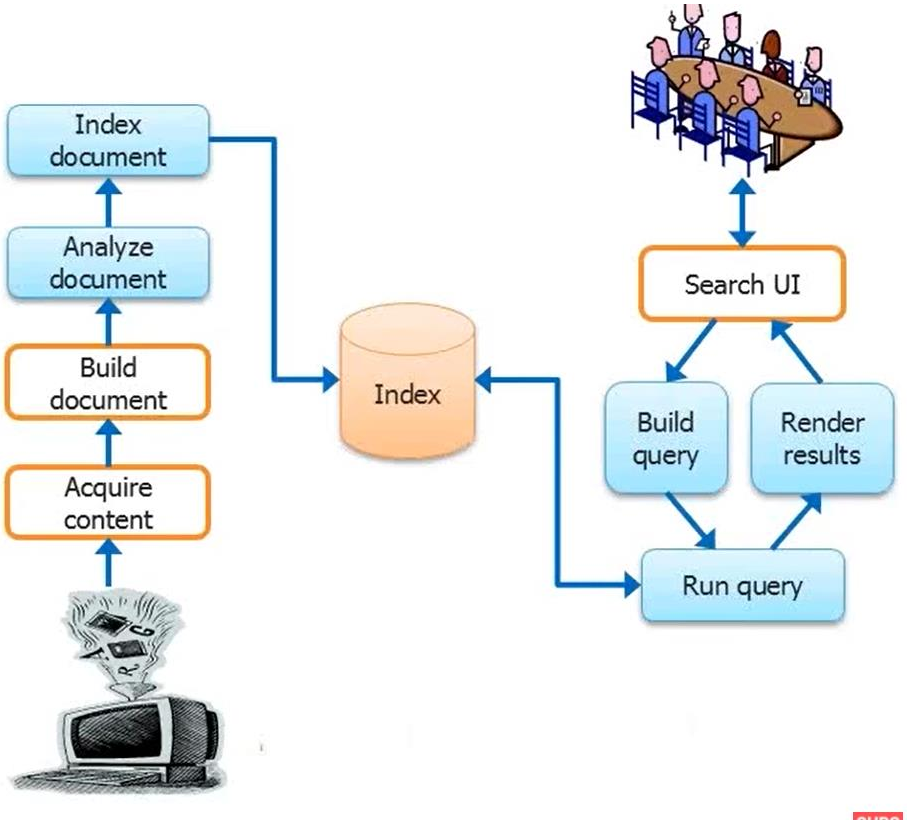
Apache Lucene

→ The first step of all search engines, is a concept called indexing

→ Indexing: is the processing of original data into a highly efficient cross-reference lookup in order to facilitate rapid searching

→ Analyze: Search engine does not index text directly. The text are broken into a series of individual atomic elements called tokens

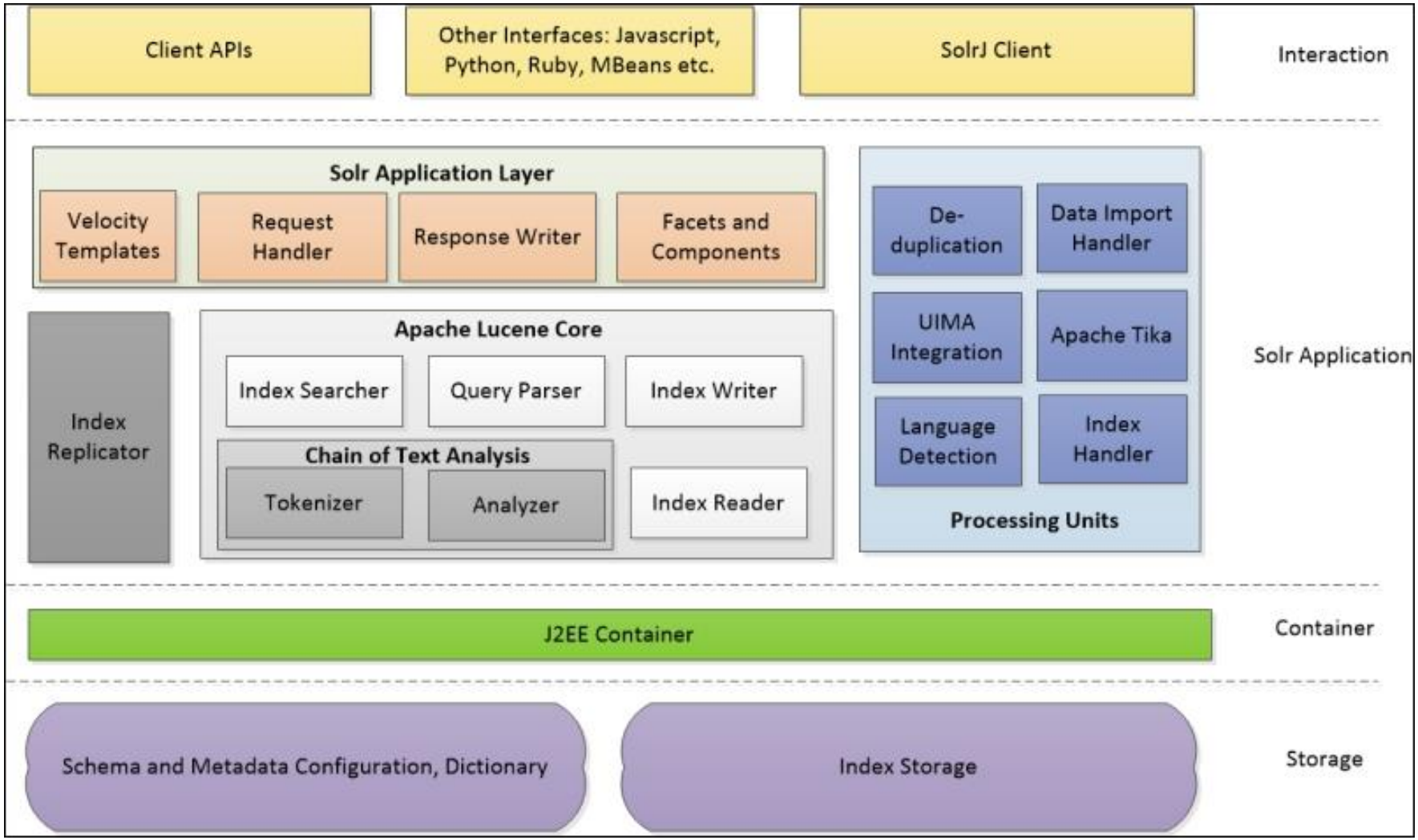
→ Searching: is the process of consulting the search index and retrieving the documents matching the query, sorted in the requested sort order



What Is Apache Solr?

- Apache Solr (stands for Searching On Lucene w/ Replication) is a free, open-source search engine based on the Apache Lucene library.
- An Apache Lucene subproject, it has been available since 2004.
- It is one of the most popular search engines available today worldwide.
- Solr is more than a search engine — it's also often used as a document-based NoSQL database with transactional support that can be used for storage purposes and even a key-value store.

Apache Solr


Client APIs
Other Interfaces: Javascript, Python, Ruby, MBeans etc.
SolrJ Client
Interaction
Solr Application Layer
Velocity Templates
Request Handler
Response Writer
Facets and Components
Index Replicator
Apache Lucene Core
Index Searcher
Query Parser
Index Writer
Chain of Text Analysis
Tokenizer
Analyzer
Index Reader
De-duplication
Data Import Handler
UIMA Integration
Apache Tika
Language Detection
Index Handler
Processing Units
Solr Application
J2EE Container
Container
Schema and Metadata Configuration, Dictionary
Index Storage
Storage

What Is Apache Solr?

- Written in Java, Solr has RESTful XML/HTTP and JSON APIs.
- Client libraries for many programming languages such as Java, Phyton, Ruby, C#, PHP, and many more being used to build search-based and big data analytics applications for websites, databases, files, etc.
- Solr takes in structured, semi-structured, and unstructured data from various sources, stores and indexes it.
- Makes it available for search in near real-time.
- Solr is also used for its analytical capabilities as well, enabling you to do faceted product search, log/security event aggregation, social media analysis, and so on.

ELK Stack

- Elastic search is a search server based on Apache Lucene. It provides real-time, distributed, multitenant-capable, full-text search engine capability.
- It provides a RESTful API using JSON documents.
- It can be used for full-text search, structured search, analytics, or a combination of all three.
- Elastic search is developed in Java and is released as open source under the terms of the Apache 2.0 license.
- One of its key features is the ability to search fast by indexing the text to be searched.

Elastic vs Lucene

- Elasticsearch is built over Lucene and provides a JSON based REST API to refer to Lucene features.
- Elasticsearch provides a distributed system on top of Lucene.
- A distributed system is not something Lucene is aware of or built for.
- Elasticsearch provides this abstraction of distributed structure.
- Elasticsearch provides other supporting features like thread-pool, queues, node/cluster monitoring API, data monitoring API, Cluster management, etc.

Elastic vs Lucene

- The Elasticsearch index is a chunk of documents just like databases consist of tables in relational world.
- In order to achieve scaling we spread the Elasticsearch Indices into multiple physical nodes / servers.
- For that, we break the Elasticsearch Indices into smaller units which are called shards.

Elastic vs Lucene

- If we want to search for a specific term (for example: "Cake" or "Cookie") we'll have to go over each shard and look for it (lets put aside how shards are being located and replicated on each node).
- This operation will take a lot of time - so we need to use an efficient data structure for this search - this is where Lucene's index comes into play.
- Each Elasticsearch shard is based on the Lucene index structure and stores statistics about terms in order to make term-based search more efficient.

Elastic vs Lucene

- Bonus - Lucene's index as inverted index
- As can be seen in the example below , Lucene's index stores the original document’s content plus additional information, such as term dictionary and term frequencies, which increase searching efficiency:

Term	Document	Frequency
Cake	doc_id_1, doc_id_8	4 (2 in doc_id_1, 2 in doc_id_8)
Cookie	doc_id_1, doc_id_6	3 (2 in doc_id_1, 1 in doc_id_6)
Spaghetti	doc_id_12	1 (1 in doc_id_12)

ELK Stack

- Many search engines have been available for a long time with the ability to search on the basis of timestamp or exact values.
- So, what's the big deal about Elastic search? It differentiates by performing full text search, handling synonyms, and scoring documents by relevance.
- Moreover, it can also generate analytics and aggregation from the same data in real time.
- This is where Elastic search scores above other search engines.
- Elastic search will make you fall in love with your data.

Elastic Search Features

- ✓ search engine/ search server
- ✓ NoSQL database i.e. can’t use SQL for queries.
- ✓ Based on Apache Lucene and provides RESTful API
- ✓ Provides horizontal scalability, reliability and multenant capability for real time search
- ✓ Uses indexes to search which makes it faster

Companies Using ElasticSearch

guardian

GitHub

StumbleUpon

WIKIPEDIA
The Free Encyclopedia

SOUNDCLOUD

ELK Stack

- Netflix uses Elasticsearch to deliver millions of messages to customers on any given day across multiple channels like email, push notifications, text, voice calls, etc.
- Salesforce has built a custom plugin on top of elastic search that enables the collection of Salesforce log data, facilitating insights into organizational usage trends and user behavior.
- *The New York Times* uses Elasticsearch to put all 15 million of its articles published over the last 160 years. This enables awesome search capability on the archives.

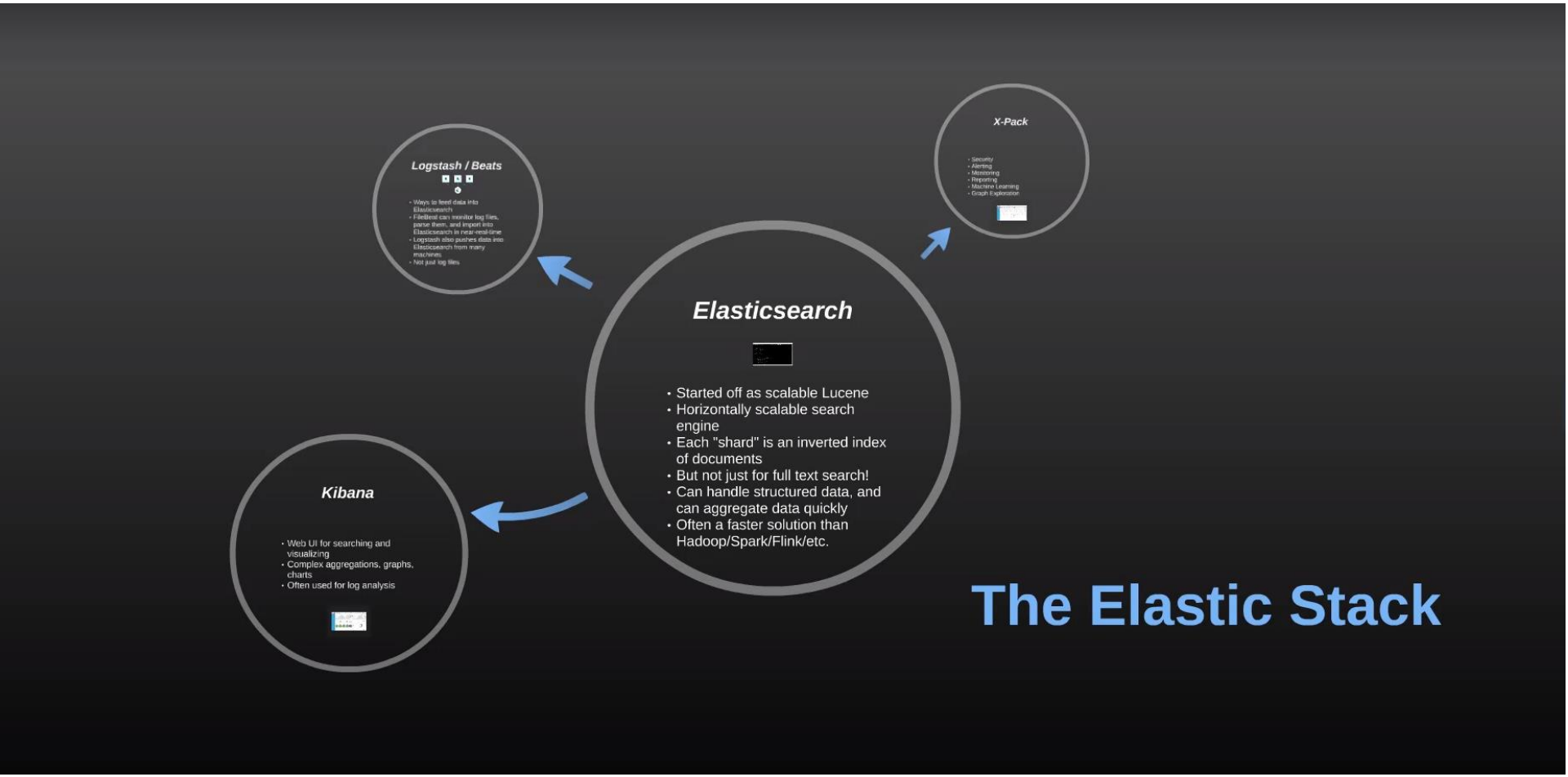
ELK Stack

- Microsoft is using Elasticsearch for search and analytics capabilities across various products like MSN, Microsoft Social Listening, and Azure Search.
- EBay has used Elasticsearch to build a flexible search platform and is further leveraging it for data analytics.
- **The Guardian**: This uses Elasticsearch to process 40 million documents per day, provide real-time analytics of site-traffic across the organization, and help understand audience engagement better.
- **StumbleUpon**: This uses Elasticsearch to power intelligent searches across its platform and provide great recommendations to millions of customers.

ELK Stack

- **Sound Cloud**: This uses Elastic search to provide real-time search capabilities for millions of users across geographies.
- **GitHub**: This uses Elastic search to index over 8 million code repositories, and index multiple events across the platform, hence providing real-time search capabilities across it.

Elastic Stack



ELK Stack

Key features of Elastic search:

- It provides real-time search and analytics of your data.
- Elastic search is a truly distributed system and can run from a humble laptop to thousands of nodes.
- It can be deployed as highly available clusters with support for multitenancy.
- Multitenancy is a software architecture in which a single instance of an application or service supports multiple customers (tenants) while ensuring privacy and security for these customers.

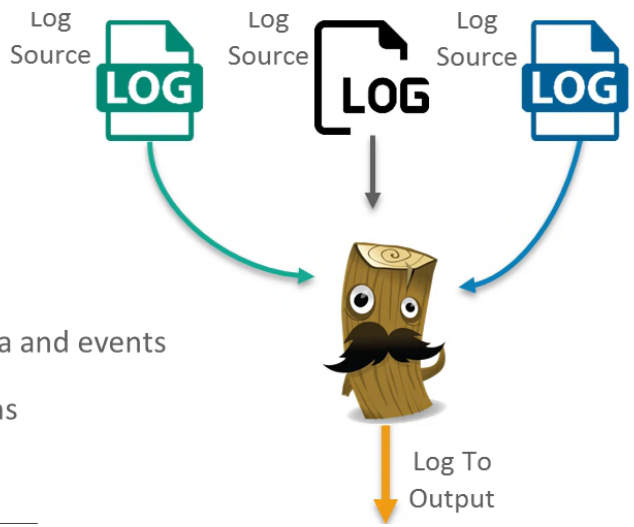
ELK Stack

- Upon the addition of a new node or failure of a node, it reorganizes and rebalances data automatically.
- Elasticsearch provides a user-friendly RESTful interface using JSON over HTTP.
- All data or information is stored as structured JSON documents.
- Elasticsearch is built on top of Apache Lucene and is available as open source software under the Apache 2 license.

Logstash

Features

- ✓ Data pipeline tool
- ✓ Centralizes the data processing
- ✓ Collects, parses and analyzes large variety of structured/ unstructured data and events
- ✓ Provides plugins to connect to various types of input sources and platforms

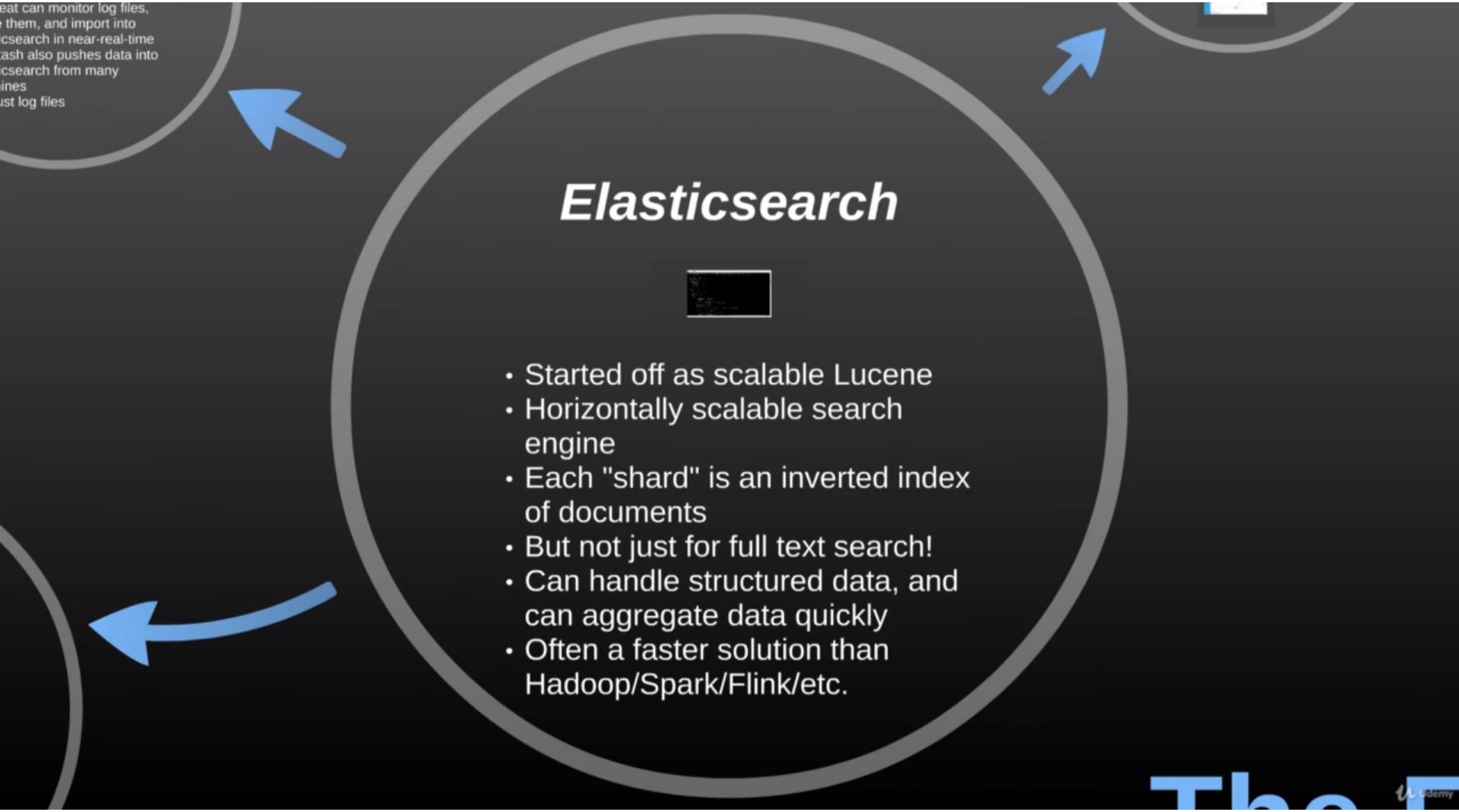


Kibana

Features

- ✓ Visualization tool
- ✓ provides real-time analysis, summarization, charting, and debugging capabilities.
- ✓ Provides instinctive and user friendly interface
- ✓ Allows sharing of snapshots of the logs searched through.
- ✓ Permits saving the dashboard and managing multiple dashboards

Elastic Search



Logstash

- Ways to feed data into Elasticsearch
- FileBeat can monitor log files, parse them, and import into Elasticsearch in near-real-time
- Logstash also pushes data into Elasticsearch from many machines
- Not just log files

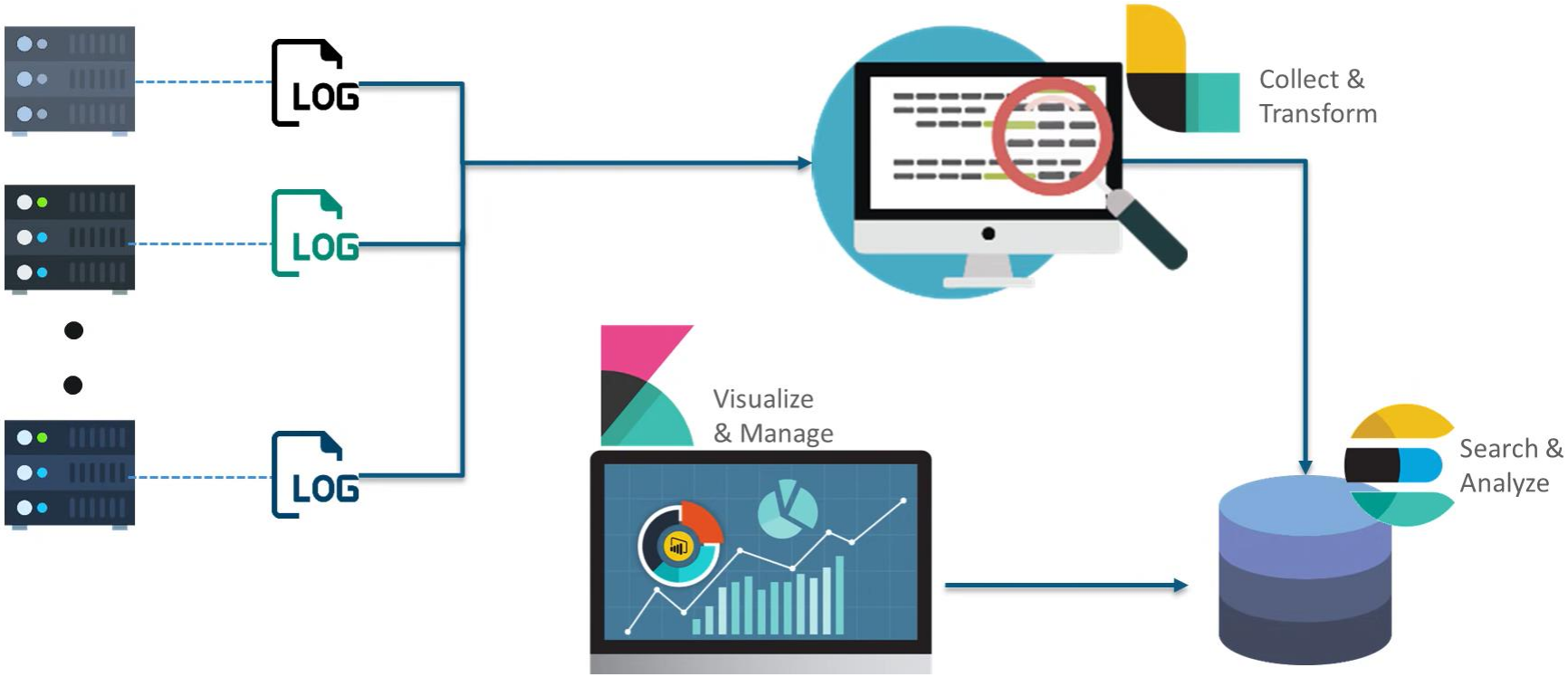
Kibana


Kibana
• Web UI for searching and visualizing
• Complex aggregations, graphs, charts
• Often used for log analysis

X-Pack


X-Pack
• Security
• Alerting
• Monitoring
• Reporting
• Machine Learning
• Graph Exploration

How ELK Stack Works?



Companies Using ELK Stack

NETFLIX

stackoverflow

accenture
High performance. Delivered.

LinkedIn

Medium

tripwire

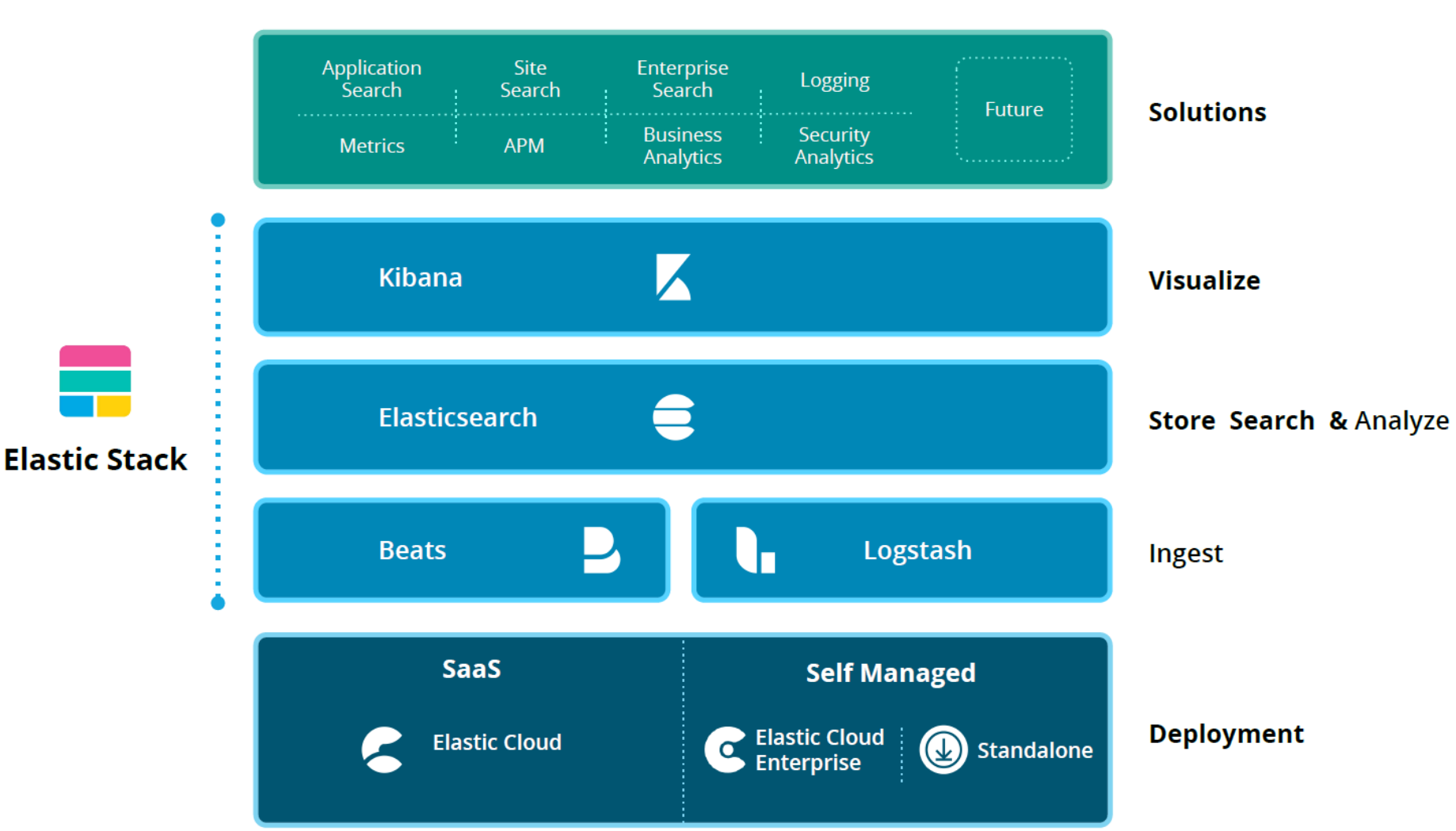
IFTTT

openstack
CLOUD SOFTWARE

HipChat

SWAT.IO

Elastic Architecture

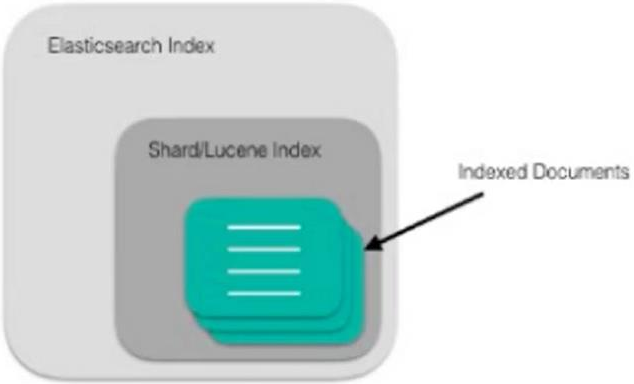


Elastic Search Basic Concepts

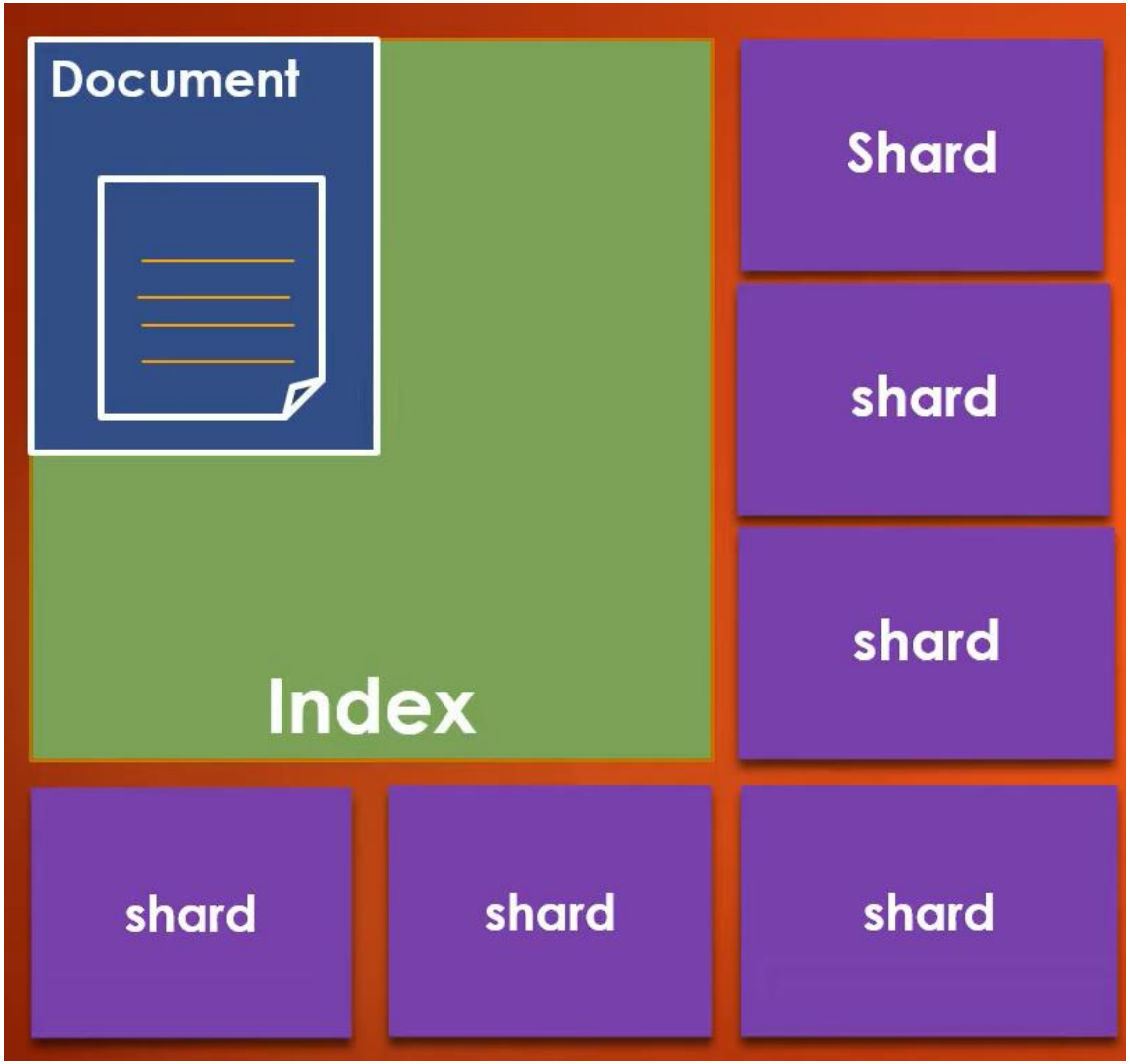
- Cluster
- Node
- Shard
- Replicas
- Index
- Documents
- Mapping
- Schemas

Node Structure

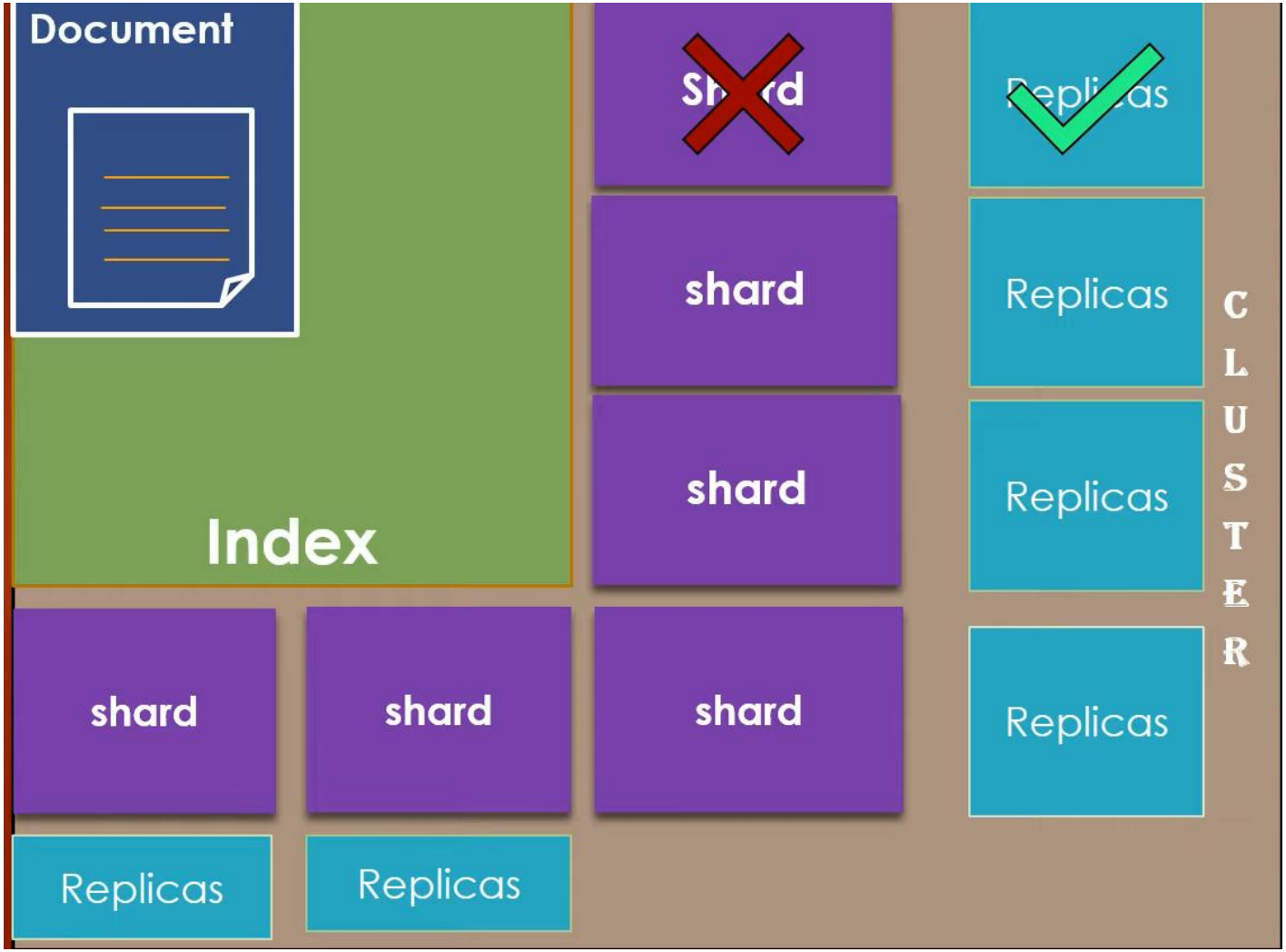
- **Index** - Logical Namespace of collection of documents
- **Shard** - Horizontal Partition of an Index
  - Eg Documents 1-10 in one shard, 11-20 in other and so on.
  - In Elasticsearch, each Shard is a self-contained Lucene index in itself.



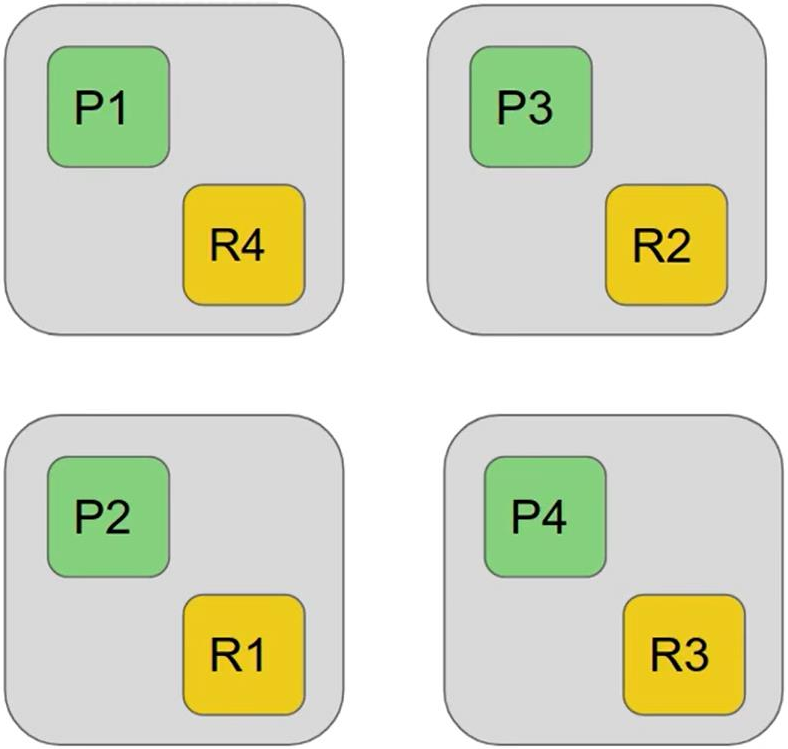
Elastic Search Concepts - Data



Elastic Search Concepts - Data

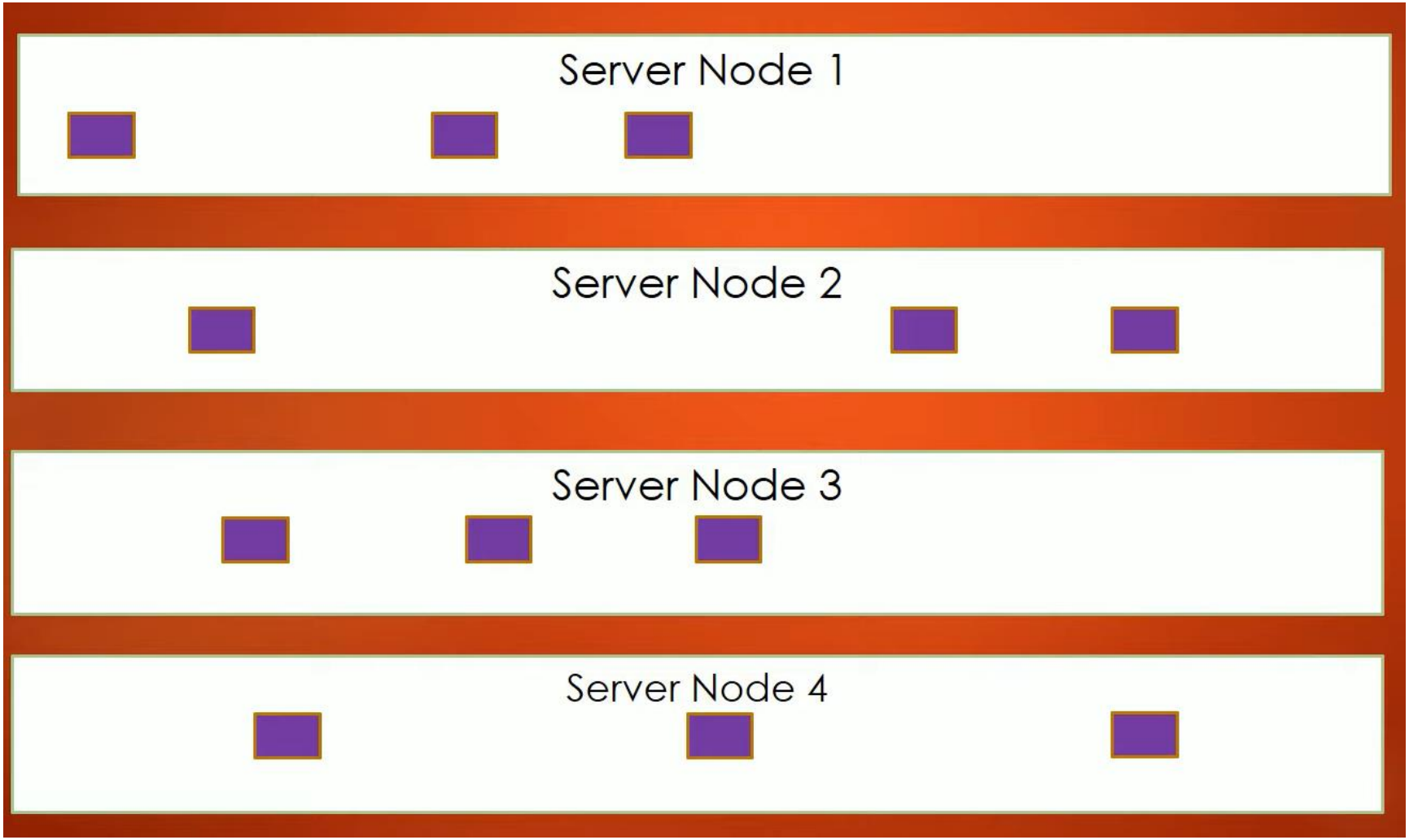


Cluster Structure



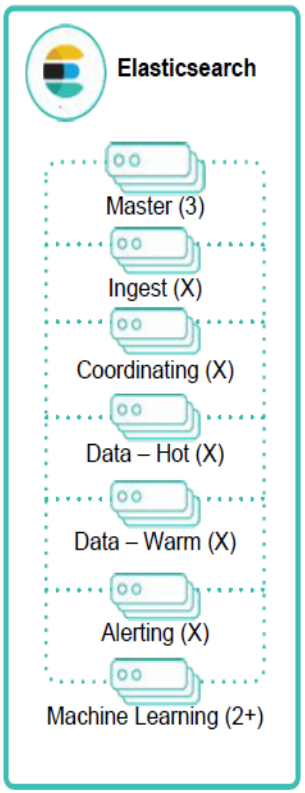
- Here we can see a cluster of 4 nodes
- Each node has 2 shards
- **Primary** and **Replica** shards
- For robustness and **fault tolerance**, each shard is replicated
- Even if a node goes down, and a primary shard is lost, a replica can be made primary until recovery
- Number of replica shards has to be set at the time of cluster creation
- Write operations on Primary and repeated on replicas and read from either

Elastic Search Concepts - Data



Elasticsearch Node Types

Nodes can play one or more roles, for workload isolation and scaling



- Master Nodes
  - Control the cluster, requires a minimum of 3, one is active at any given time
- Data Nodes
  - Hold indexed data and perform data related operations
  - Differentiated Hot and Warm Data nodes can be used
- Ingest Nodes
  - Use ingest pipelines to transform and enrich before indexing
- Coordinating Nodes
  - Route requests, handle search reduce phase, distribute bulk indexing
  - All nodes function as coordinating nodes
- Alerting Nodes
  - Run alerting jobs
- Machine Learning Nodes
  - Run machine learning jobs

Elastic Node Types

- Any time that you start an instance of Elasticsearch, you are starting a node.
- A collection of connected nodes is called a cluster.
- Every node in the cluster can handle HTTP and transport traffic by default.
- The transport layer is used exclusively for communication between nodes; the HTTP layer is used by REST clients.

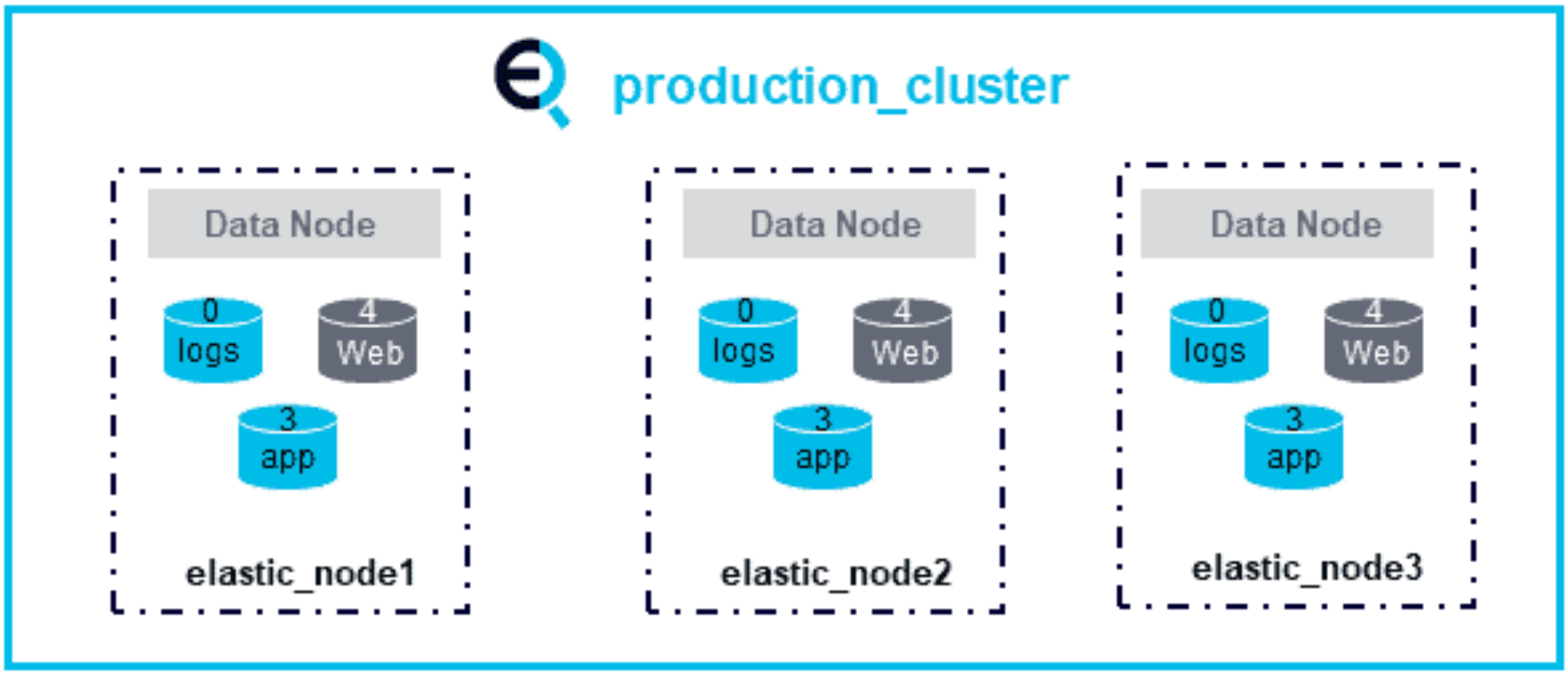
Node Roles

- Node’s roles is set using node.roles in elasticsearch.yml.
- Every cluster requires the following node roles:
  - master
  - data_content and data_hot
  - OR
  - data

Node Roles

- Master-eligible node
  - Typically, in Elasticsearch, all nodes are master-eligible by default.
  - Master nodes are responsible for certain critical tasks throughout the cluster, including creating or deleting indexes, tracking nodes, and allocating shards to nodes.
  - The master node for your cluster is elected from among your master-eligible nodes, and there can only be one node serving in the master role at a time.
- Data node
  - Data nodes are responsible for holding data and performing data-related operations.
  - This includes CRUD operations, indexing, search, and aggregations.
  - You can configure data nodes so that they only do search and aggregation, not any indexing, to reduce the load in the individual nodes.
  - All nodes are data nodes by default.

Node Roles


production_cluster
Data Node
0
logs
4
Web
3
app
elastic_node1
Data Node
0
logs
4
Web
3
app
elastic_node2
Data Node
0
logs
4
Web
3
app
elastic_node3

Node Roles

- Data content node
  - Data content nodes are part of the content tier.
  - These types of nodes will be used mainly to store archive and catalog data, where we might not do real-time indexing or frequent indexing like logs.
  - Even though these types of data will not be indexed frequently, their requirement would be to fetch results faster.
  - To provide better search performance, these types of nodes are optimized.
  - They prioritize query processing over usual I/O throughput, so complex searches and aggregations will be processed quickly

Node Roles

- Data hot node
  - Data hot nodes are part of the hot tier. This role is not necessary unless you want to configure hot-cold architecture.
  - Hot tier nodes are mainly used to store the most frequently updated and recent data. These types of data nodes should be fast during both search and indexing. Therefore, they require more RAM, CPU and fast storage.
  - To set this node role, edit the node's "elasticsearch.yml" and add the following line:
  - node.roles: ["data_hot"]

Node Roles

- Data warm node
  - Data warm nodes are part of the warm tier.
  - This role is not necessary unless you want to configure hot-cold architecture.
  - Warm tier nodes are used for storing time series data that are less frequently queried and rarely updated.
  - Warm nodes will typically have larger storage capacity in relation to their RAM and CPU.

Node Roles

- Data cold node
  - Data cold nodes are part of the cold tier. This role is not necessary unless you want to configure hot-cold architecture.
  - Time series data that no longer needs to be searched regularly will be moved from the warm tier to the cold tier.
  - Since search performance is not a priority, these nodes are usually configured to have higher storage capacity for a given RAM and CPU.

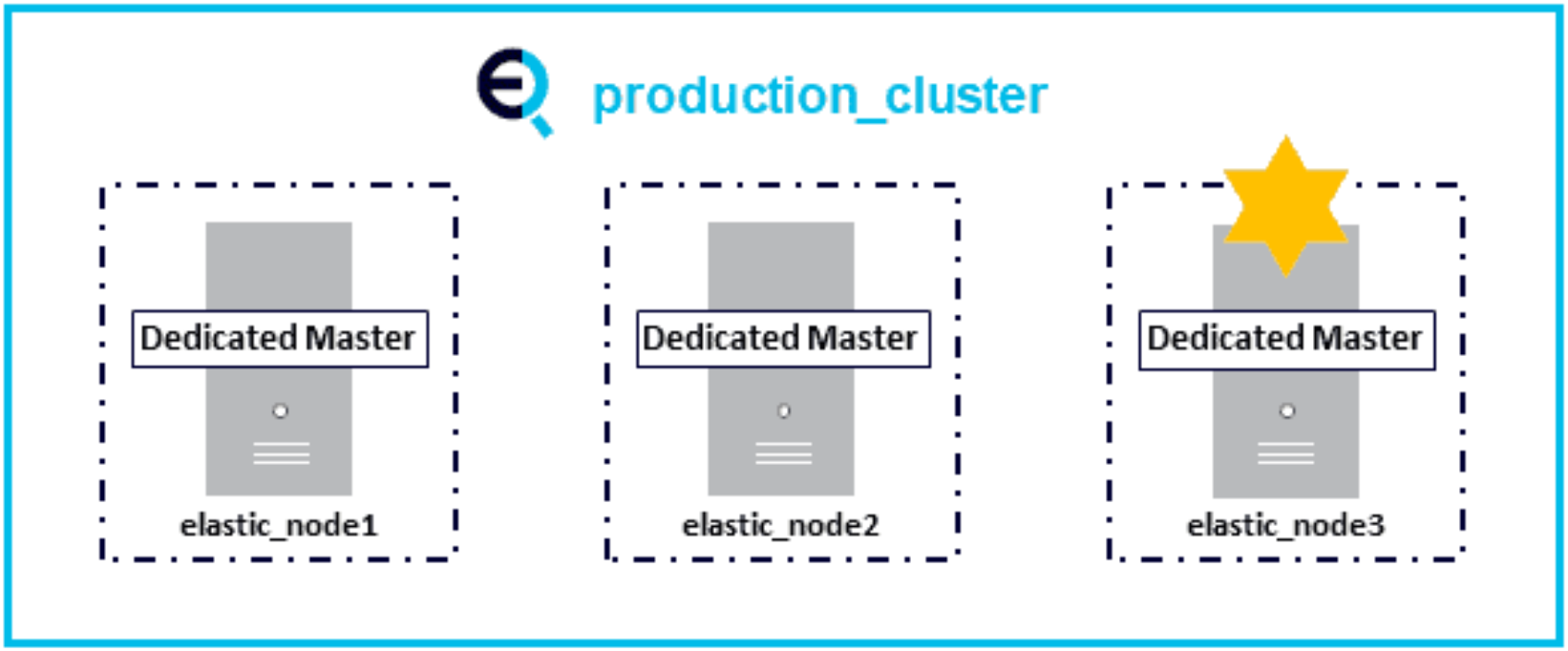
Node Roles

- Data frozen node
  - Data frozen nodes are part of the frozen tier. This role is not necessary unless you want to configure hot-cold architecture.
  - Data that is queried rarely and never updated will be moved from cold tier to the frozen tier.
  - This type of node may reduce storage and operating costs, while still allowing the user to search on frozen data.

Node Roles

- Dedicated master nodes can be provisioned with fewer resources than data nodes because they only handle cluster state and not user data.
- Placing these nodes into different failure zones or availability zones of a cloud, along with multiple copies of the data on the data nodes, enables the cluster to survive numerous types of server, zone, and data center failures.

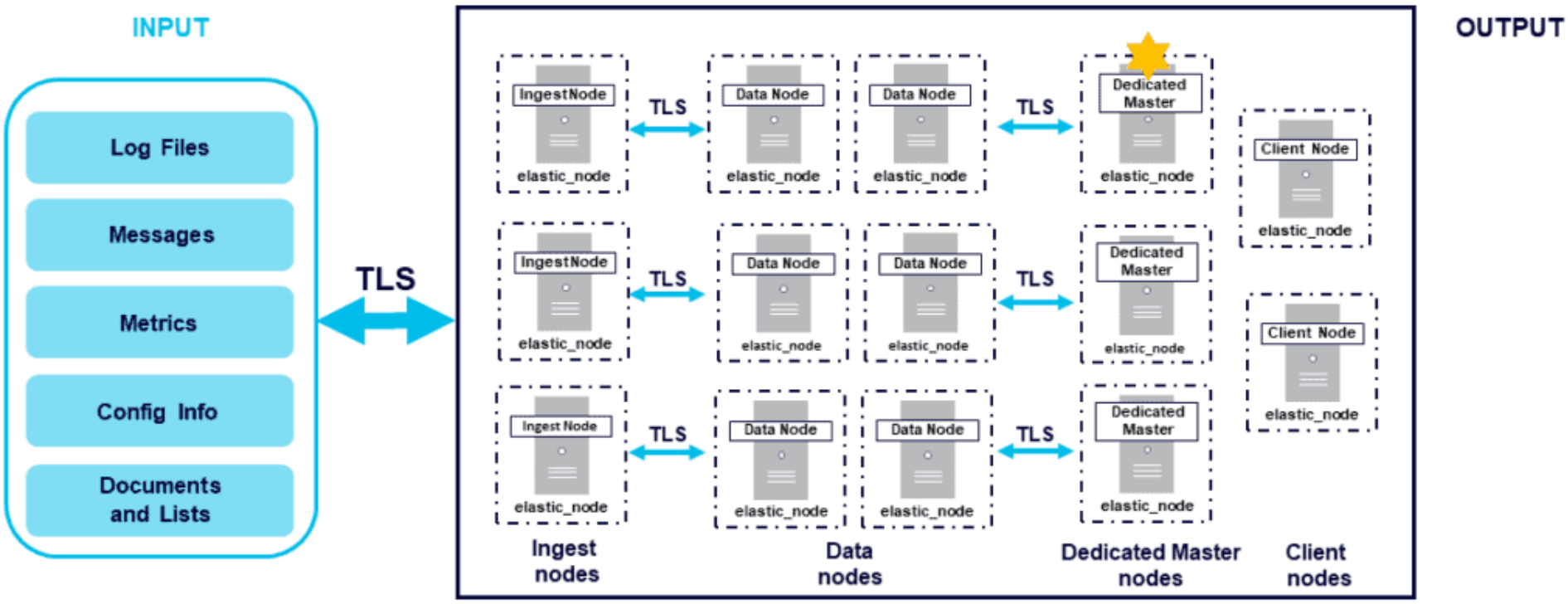
Node Roles


production_cluster
Dedicated Master
elastic_node1
Dedicated Master
elastic_node2
Dedicated Master
elastic_node3

Node Roles

- Ingest node
  - Ingest nodes oversee pre-processing documents before they are indexed.
  - These are also known as “transform” nodes because they help transform documents for indexing.
  - All nodes are also ingesting nodes by default.
  - Some organizations elect to use ingest nodes instead of Logstash for piping in and processing log data.

Node Roles


INPUT
Log Files
Messages
Metrics
Config Info
Documents and Lists
TLS
IngestNode
Ingest Node
Data Node
Dedicated Master
Client Node
elastic_node
TLS
Ingest nodes
Data nodes
Dedicated Master nodes
Client nodes
OUTPUT

Node Roles

- Remote-eligible node
  - A node that has the remote_cluster_client role, which makes it eligible to act as a remote client.
- Machine learning node
  - A node that has the ml role.
  - If you want to use machine learning features, there must be at least one machine learning node in your cluster.
  - For more information, see Machine learning settings and Machine learning in the Elastic Stack.

Node Roles

- Transform node
  - A node that has the transform role.
  - If you want to use transforms, there must be at least one transform node in your cluster.
- Coordinating Nodes
  - Coordinating-only nodes act as load-balancers. This type of node routes requests to data nodes and handles bulk indexing by distributing the requests.
  - These types of nodes are used in larger clusters. By getting the cluster state from all the nodes, the coordinating-only node will route requests accordingly.

Node Roles

- Coordinating Nodes
  - In small clusters, it is usually not necessary to use a coordinating node, since the same role will be handled by data nodes, and the greater complexity is not justified on a small cluster.
  - In the scatter phase, the coordinating node forwards the request to the data nodes which hold the data.
  - Each data node executes the request locally and returns its results to the coordinating node.
  - In the gather phase, the coordinating node reduces each data node’s results into a single global result set.

Node Roles

- Coordinating Nodes
  - Every node is implicitly a coordinating node.
  - This means that a node that has an explicit empty list of roles via node.roles will only act as a coordinating node, which cannot be disabled.
  - As a result, such a node needs to have enough memory and CPU in order to deal with the gather phase.

Node Roles

- Voting-only master-eligible
  - A voting-only master-eligible node is a node that participates in master elections but which will not act as the cluster's elected master node.
  - In particular, a voting-only node can serve as a tiebreaker in elections.
  - To configure a master-eligible node as a voting-only node, include master and voting_only in the list of roles. For example to create a voting-only data node:
    - node.roles: [data, master, voting_only]

Node Roles

- Voting-only master-eligible
  - High availability (HA) clusters require at least three master-eligible nodes, at least two of which are not voting-only nodes.
  - Such a cluster will be able to elect a master node even if one of the nodes fails.

Node Details

- # return just indices
- http://localhost:9200/_nodes/stats/indices
- # return just os and process
- http://localhost:9200/_nodes/stats/os,process
- # return just process for node with IP address 127.0.0.1
- http://localhost:9200/_nodes/127.0.0.1/stats/process

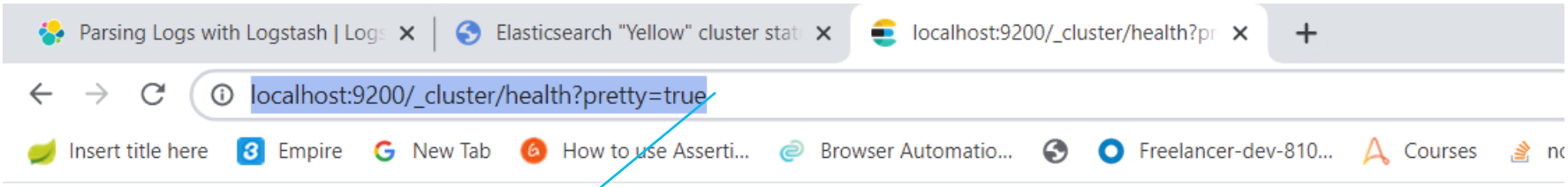
Cluster Health API

- http://localhost:9200/_cluster/health?pretty=true
- Elasticsearch provides a handy "traffic lights" classification of cluster health. Here is a simple explanation of each of the options.
  - RED: Some or all of (primary) shards are not ready.
  - YELLOW: Elasticsearch has allocated all of the primary shards, but some/all of the replicas have not been allocated.
  - GREEN: Your cluster is fully operational. Elasticsearch is able to allocate all shards and replicas to machines within the cluster.

Cluster Health API

- Currently, our cluster health is yellow, meaning shard replicas have not been allocated.
- This is because the current cluster only consists of a single node, so the replicas remain unassigned simply because no other node is available to contain them.
- We can fix this by adding another node to the cluster

Cluster Health API




```
{
  "cluster_name" : "elasticsearch",
  "status" : "yellow",
  "timed_out" : false,
  "number_of_nodes" : 1,
  "number_of_data_nodes" : 1,
  "active_primary_shards" : 7,
  "active_shards" : 7,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 6,
  "delayed_unassigned_shards" : 0,
  "number_of_pending_tasks" : 0,
  "number_of_in_flight_fetch" : 0,
  "task_max_waiting_in_queue_millis" : 0,
  "active_shards_percent_as_number" : 53.84615384615385
}
```

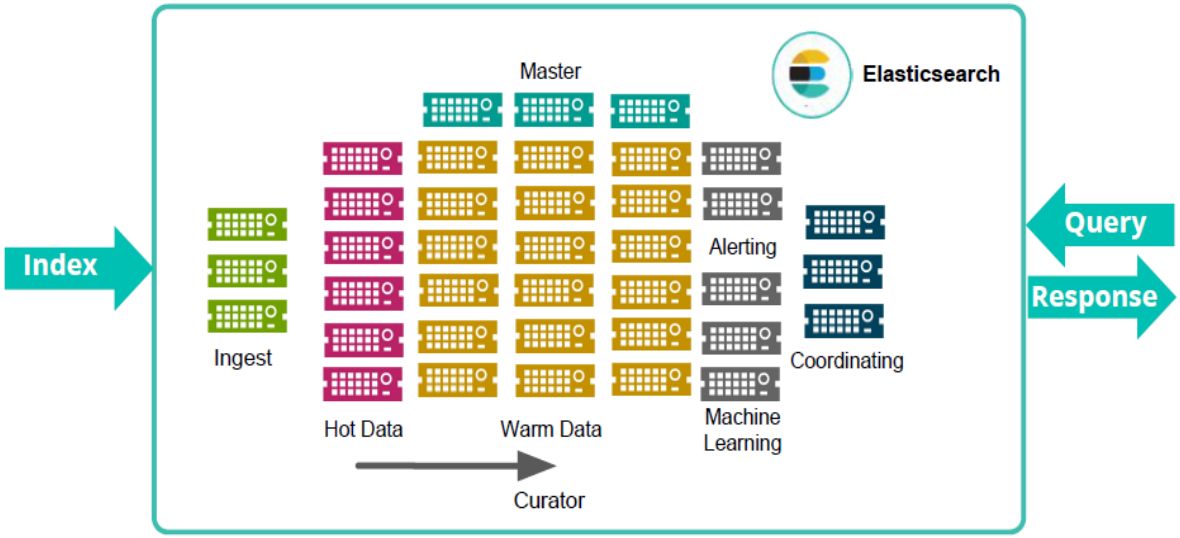

Single node cluster

Cluster Configuration

```
#give your cluster a name.

cluster.name: my-cluster

#give your nodes a name (change node number from node to node).

node.name: "es-node-1"

#define node 1 as master-eligible:

node.master: true

#define nodes 2 and 3 as data nodes:

node.data: true

#enter the private IP and port of your node:

network.host: 172.11.61.27

http.port: 9200
```

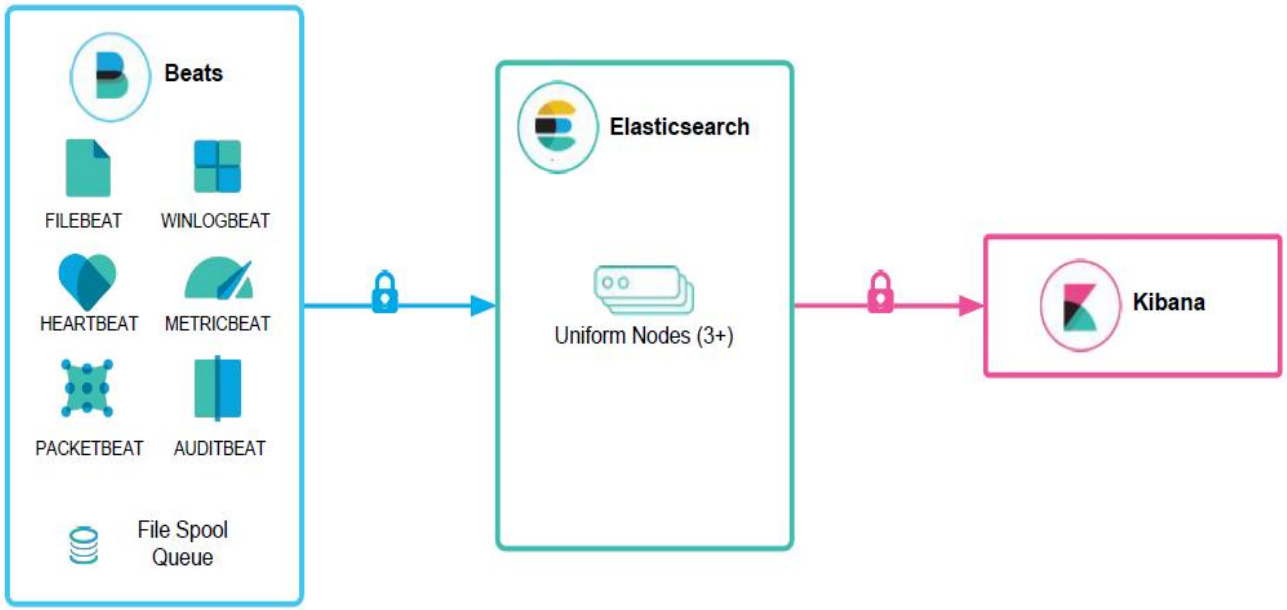
Inside a Large Elasticsearch Logging Cluster

Reduce infrastructure costs, isolate workloads, and manage data lifecycle



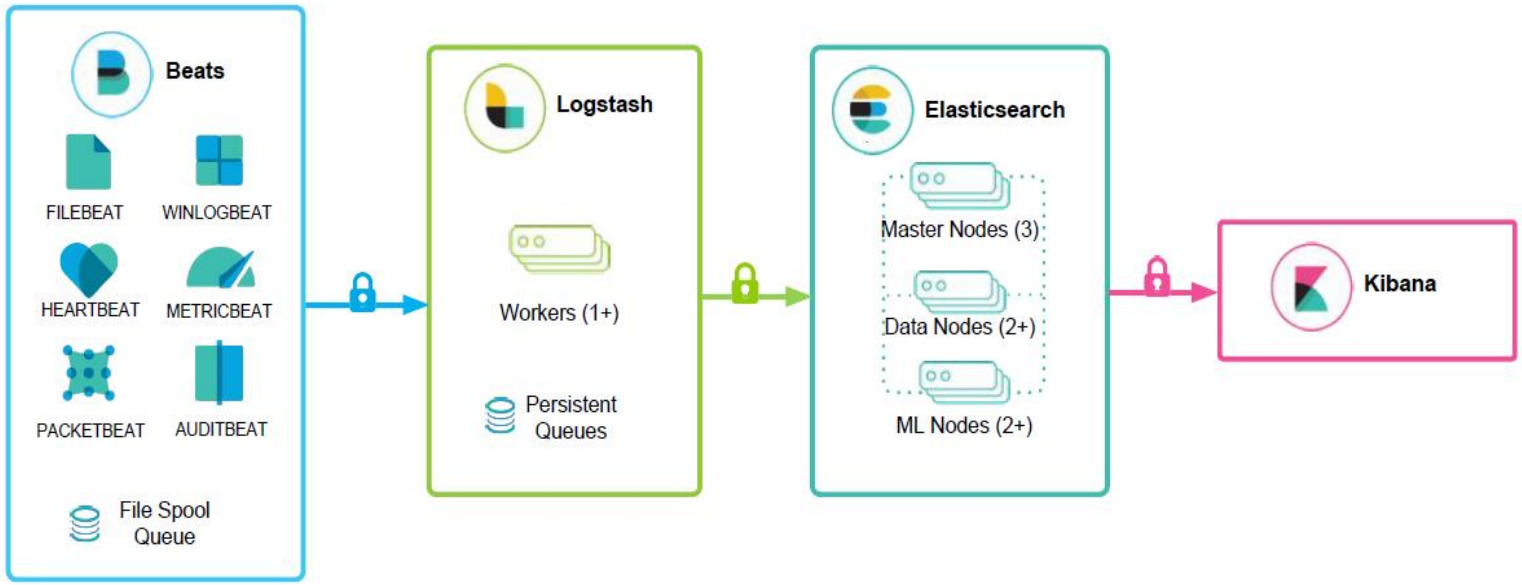
Quick Start

Beats, Elasticsearch and Kibana

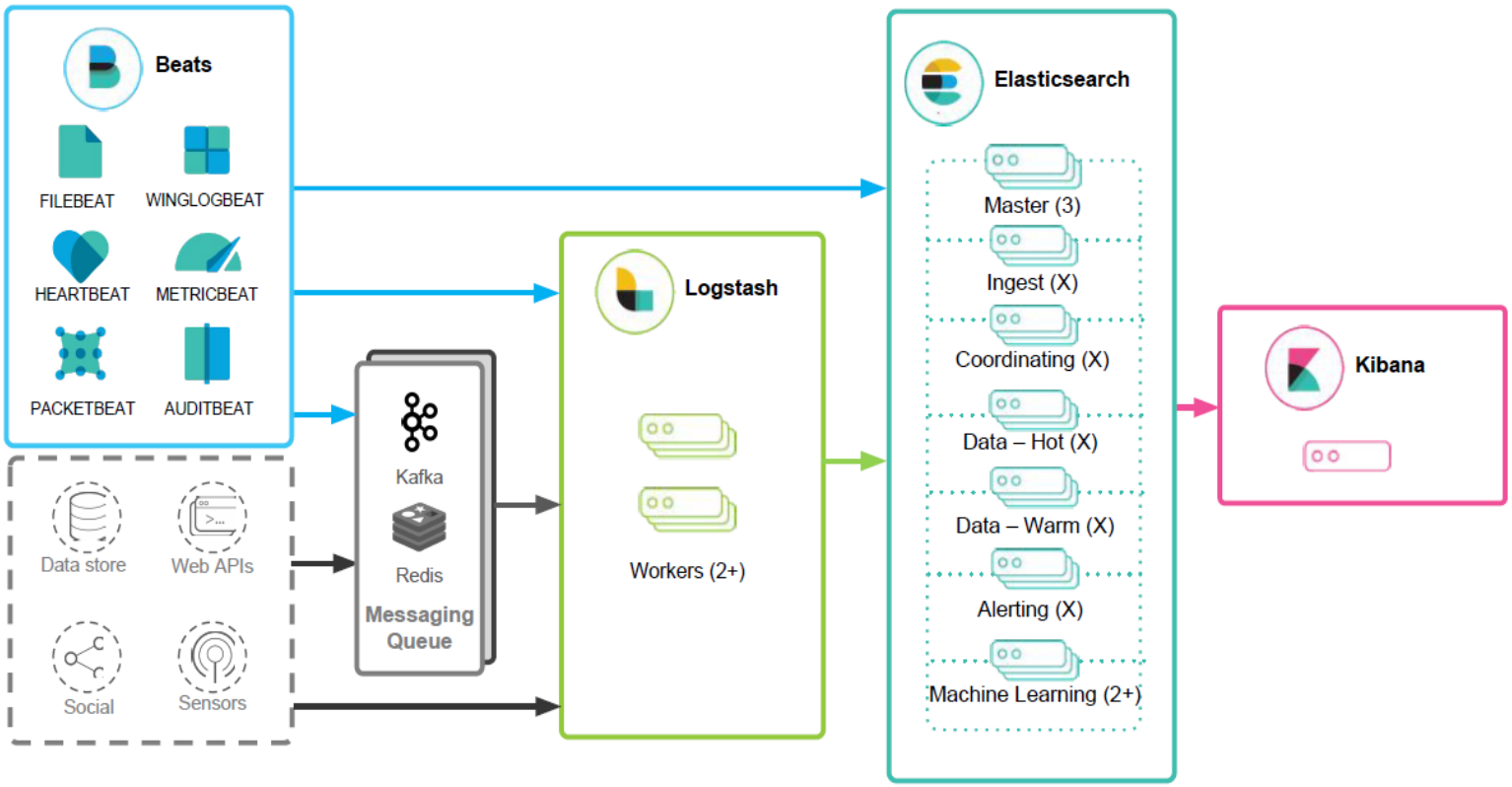


Advanced Processing and Resiliency

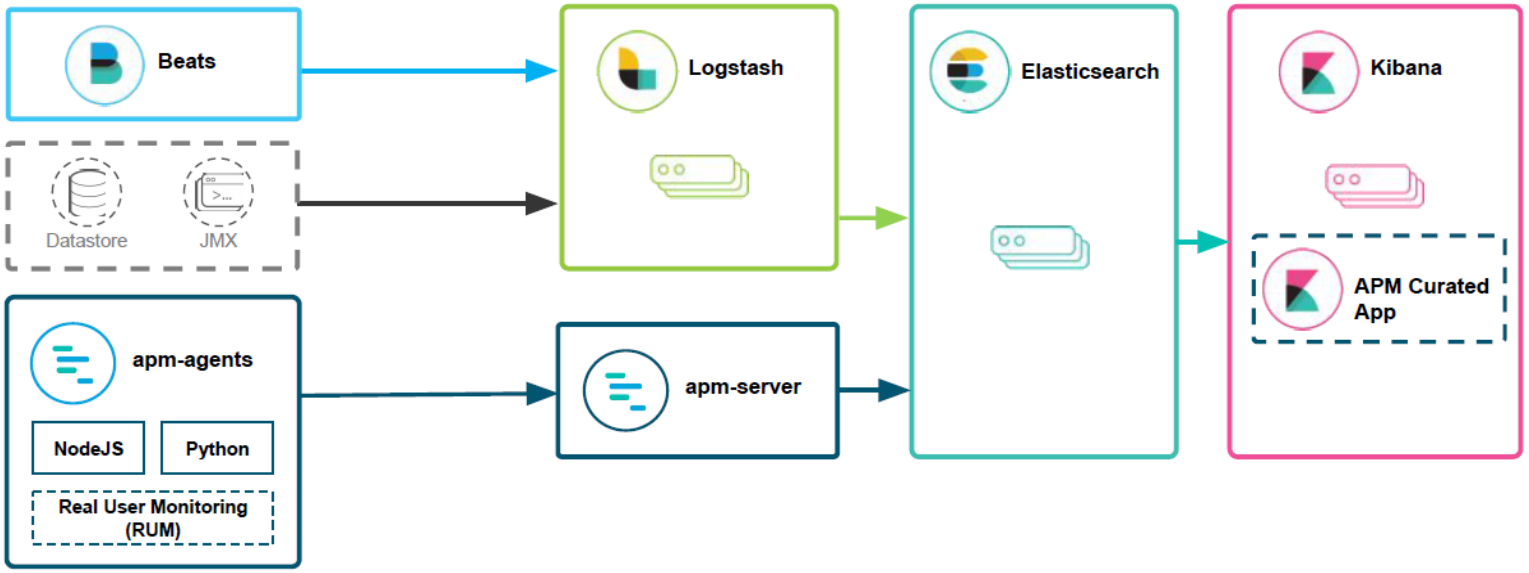
Adding Logstash processing, differentiated Elasticsearch node types



Flexible ingestion and input sources

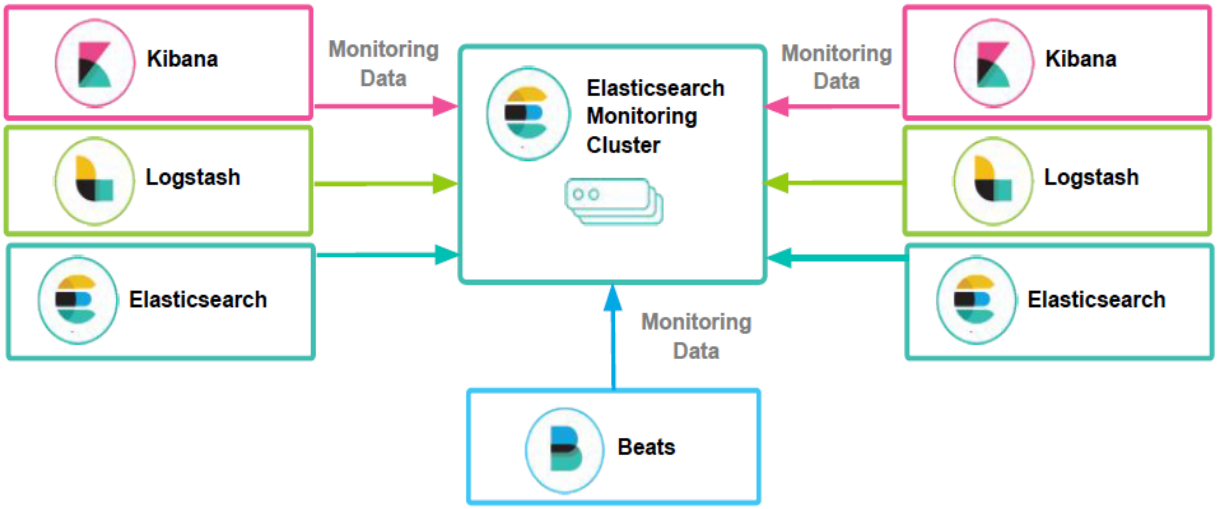


Application Metric Collection with Elastic APM



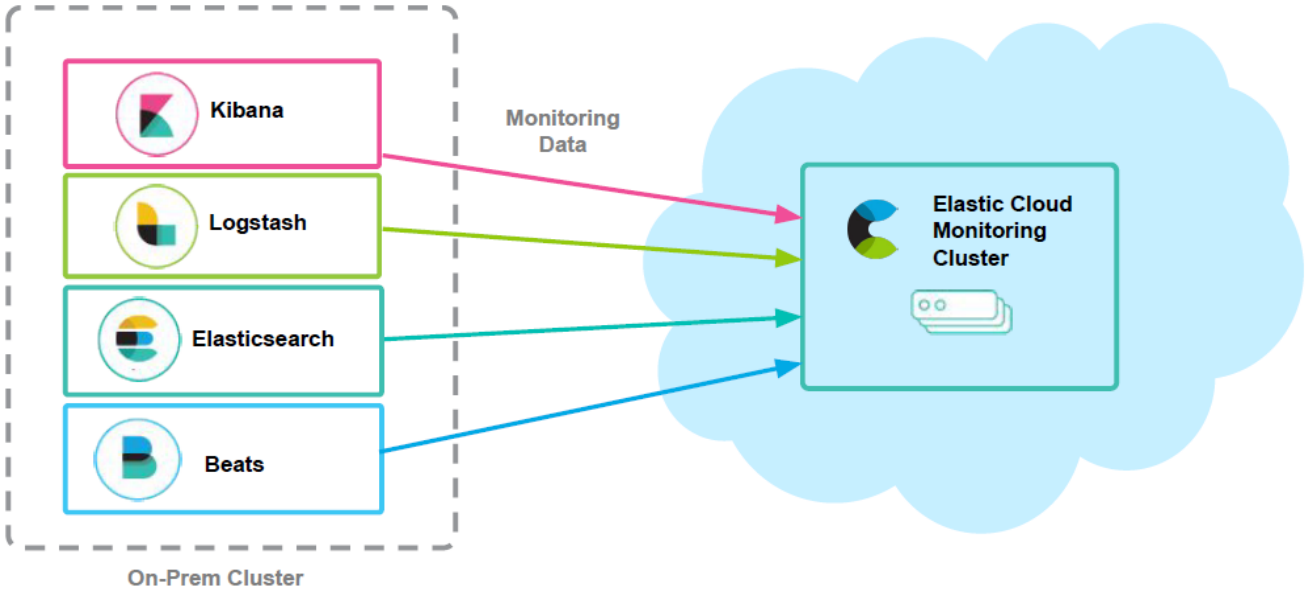
Centralized Monitoring Cluster

Maintain isolated monitoring cluster for monitoring workload isolation



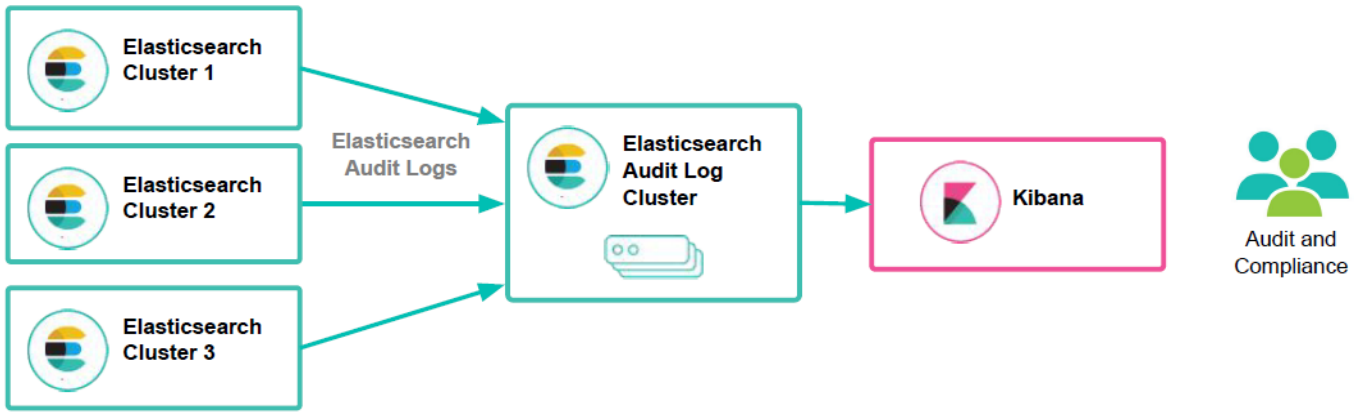
Cloud Monitoring Cluster

Opt-in Elastic Cloud cluster for monitoring on-premise stack

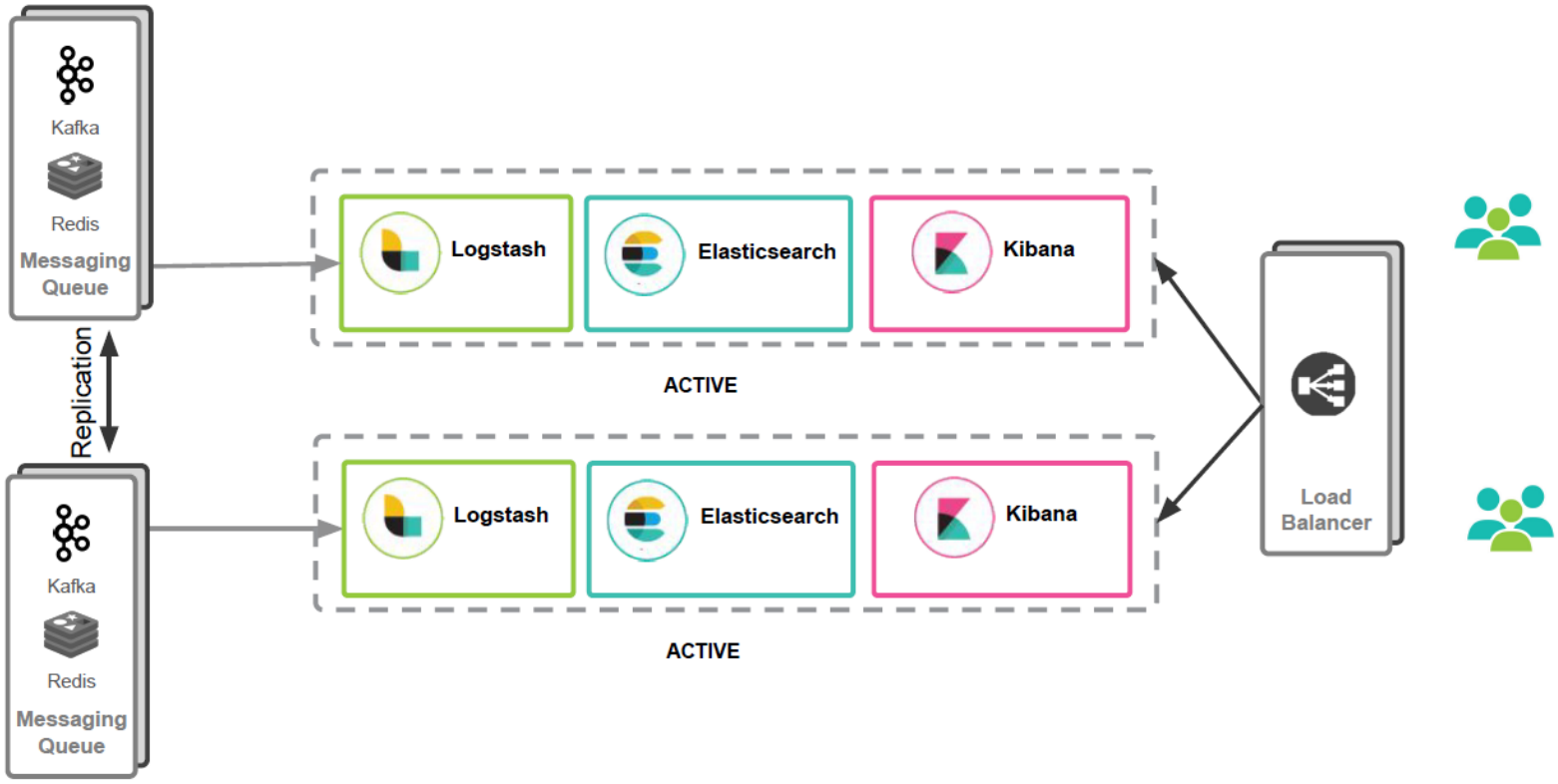


Isolated Audit Logging Cluster

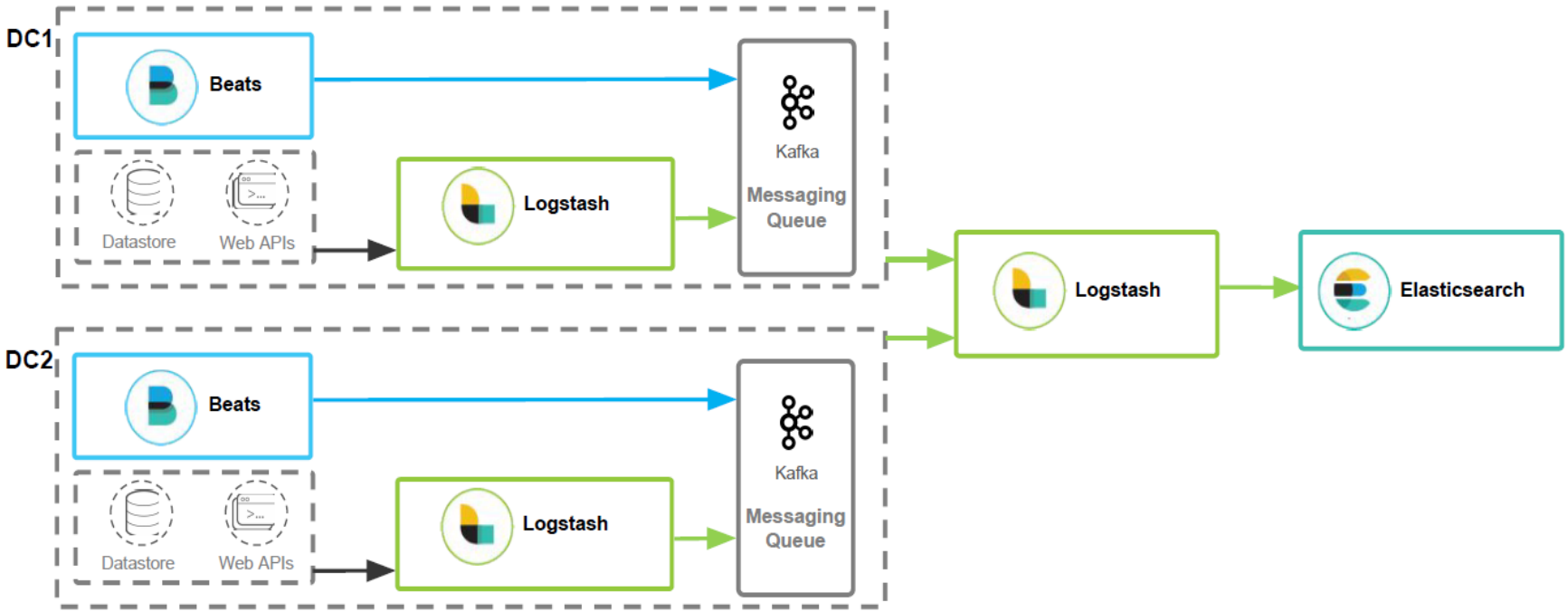
Maintain isolated audit logging cluster for increased security and compliance



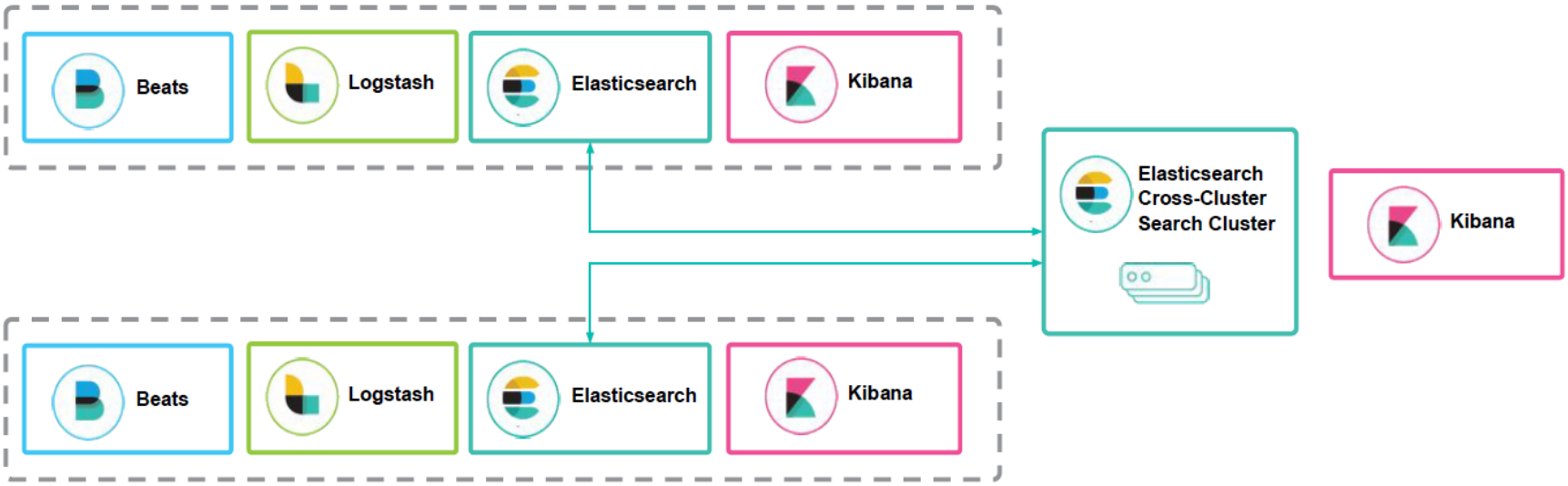
Multiple Data Centers, Duplicate Data



Multi Data Centers with a Queue at Each DC

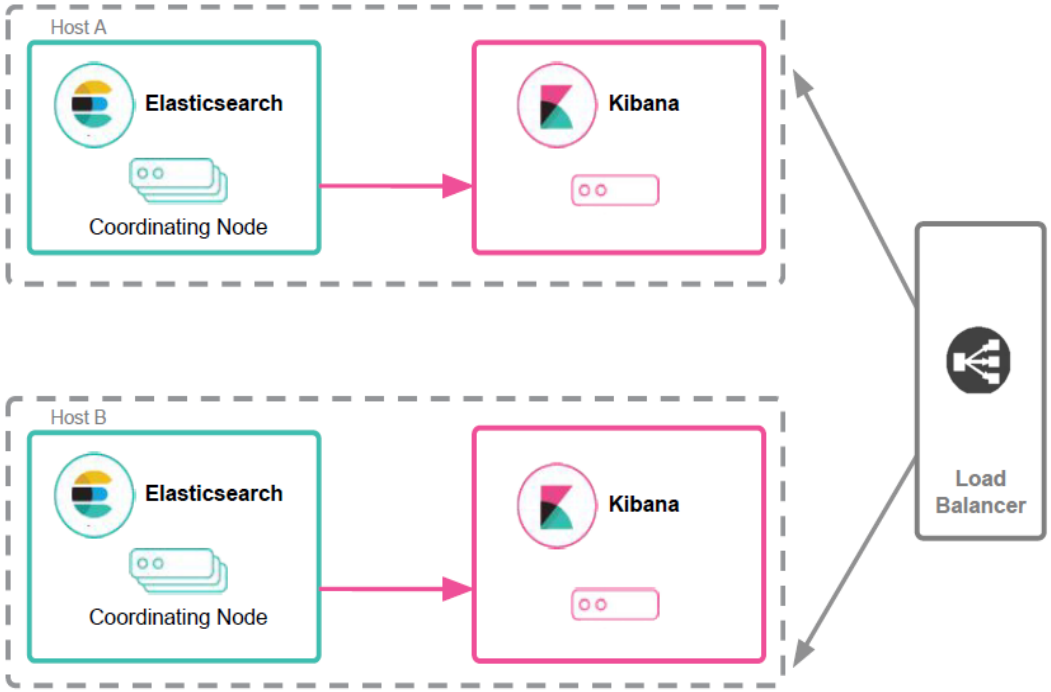


Multi Data Center, Distinct Data and Cross-Cluster Search



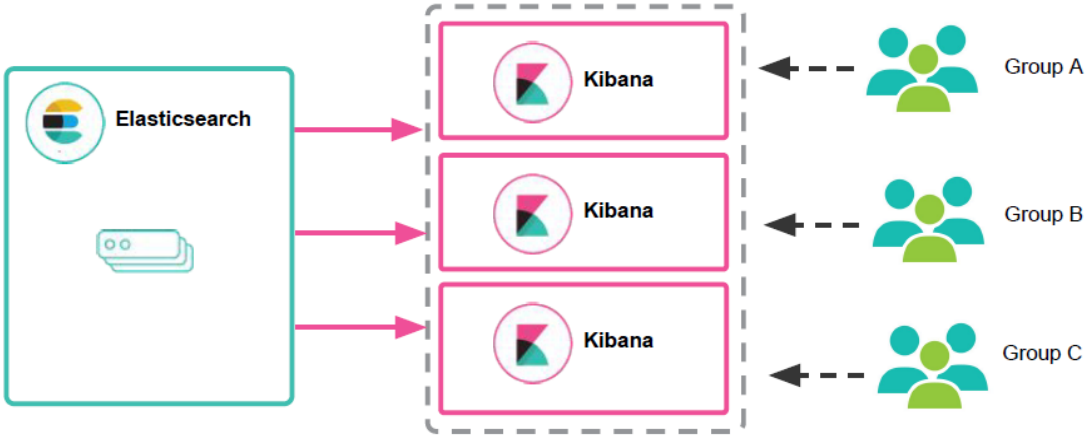
High Availability

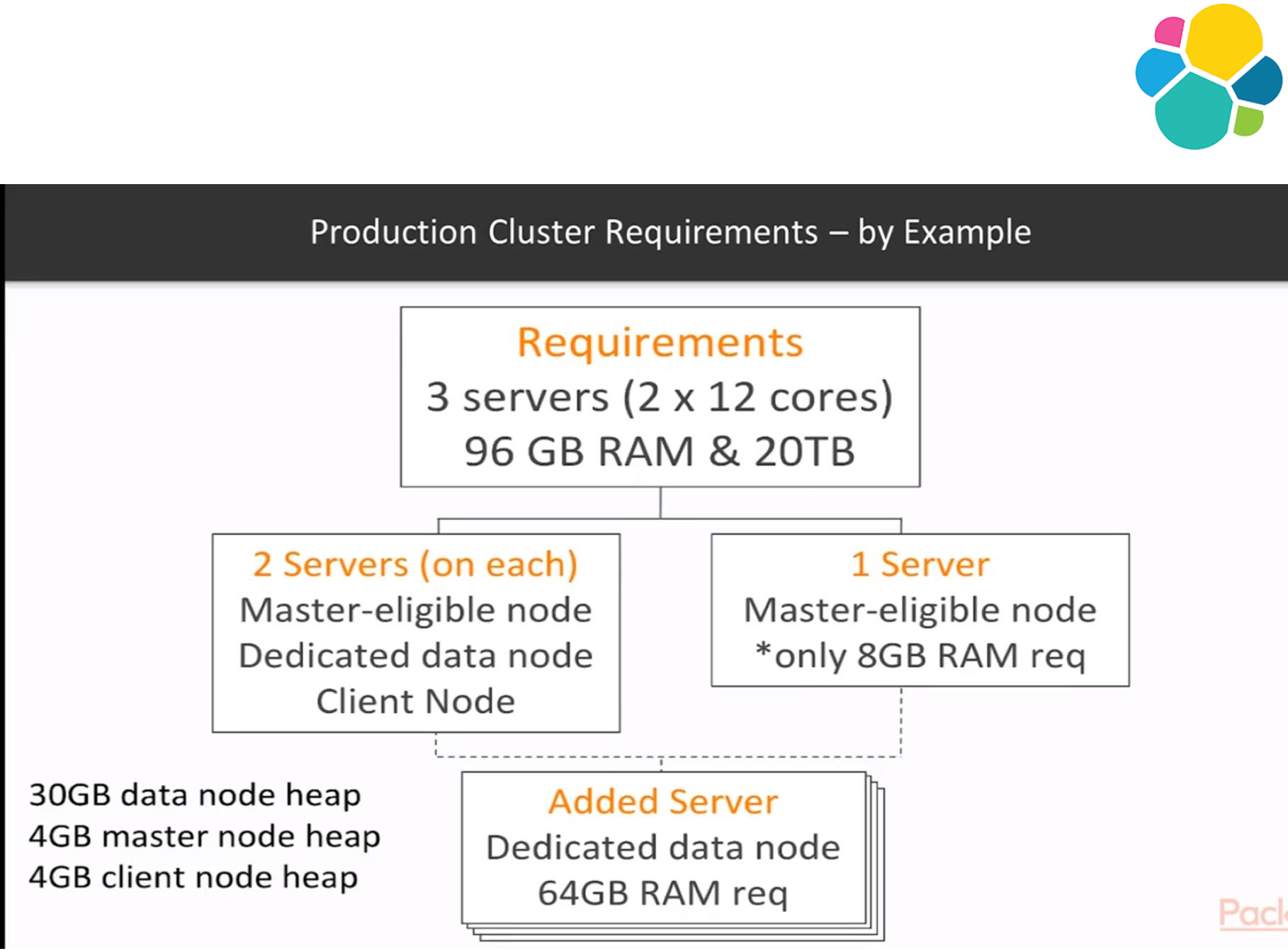
Pair two coordinating nodes with two independent Kibana nodes



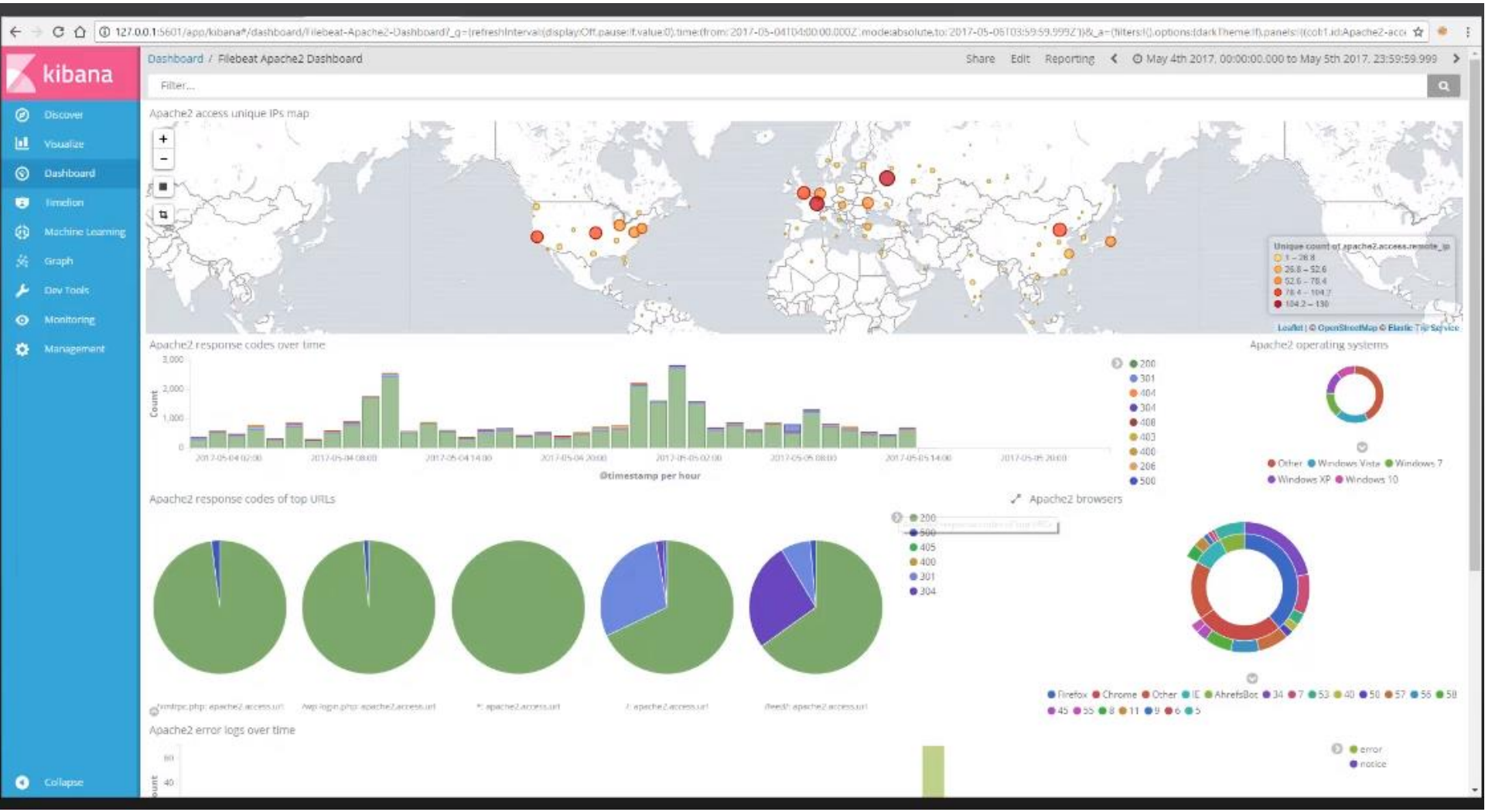
Separating Dashboards by Groups

Isolate user content by group in different Kibana instances




Production Cluster Requirements – by Example
Requirements
3 servers (2 x 12 cores)
96 GB RAM & 20TB
2 Servers (on each)
Master-eligible node
Dedicated data node
Client Node
1 Server
Master-eligible node
*only 8GB RAM req
Added Server
Dedicated data node
64GB RAM req
30GB data node heap
4GB master node heap
4GB client node heap
Packt

Kibana


kibana
Discover
Visualize
Dashboard
Timelion
Machine Learning
Graph
Dev Tools
Monitoring
Management
Collapse
Dashboard / Filebeat Apache2 Dashboard
Share
Edit
Reporting
May 4th 2017, 00:00:00.000 to May 5th 2017, 23:59:59.999
Filter...
Apache2 access unique IPs map
Apache2 response codes over time
Count
@timestamp per hour
Apache2 operating systems
Other
Windows Vista
Windows 7
Windows XP
Windows 10
Apache2 response codes of top URLs
Apache2 browsers
Firefox
Chrome
Other
IE
AhrefsBot
Apache2 error logs over time
error
notice

Logstash

- Instead of using the traditional ways of generating and analyzing logs, which has its own pitfalls, it is much better to use Logstash, which is the next generation logging framework.
- Logstash is essentially an integrated framework for log collection, centralization, parsing, storage, and search.
- It is an open source software that can dynamically unify data from disparate sources and normalize the data into destinations of your choice.

Logstash

- Open source event processing engine.
- Supports different sources and destinations.
- Can manipulate data, too
- Flexible Configuration
- Pipeline
- An Orchestration of plugins
- Plugins receive data, manipulates and sends it.
- 100’s of plugins in logstash

Logstash

Pipeline = input + (filter) + output

Each phrase uses plugins

Processed events are sent to stashes

A stash is a destination, e.g. Elasticsearch or Kafka

Logstash is not limited to processing logs!

Handles XML, CSV, JSON, etc. alike

Decoupled architecture

Centralized event processing @ Logstash

Logstash

- Logstash is a data pipeline that helps collect, parse, and analyze a large variety of structured and unstructured data and events generated across various systems.
- It provides plugins to connect to various types of input sources and platforms.
- It is designed to efficiently process logs, events, and unstructured data sources for distribution into a variety of outputs with the use of its output plugins, namely file, stdout (as output on console running Logstash), or Elasticsearch.

Logstash

- It has the following key features:
- **Centralized data processing**: Logstash helps build a data pipeline that can centralize data processing.
- With the use of a variety of plugins for input and output, it can convert a lot of different input sources to a single common format.

Logstash

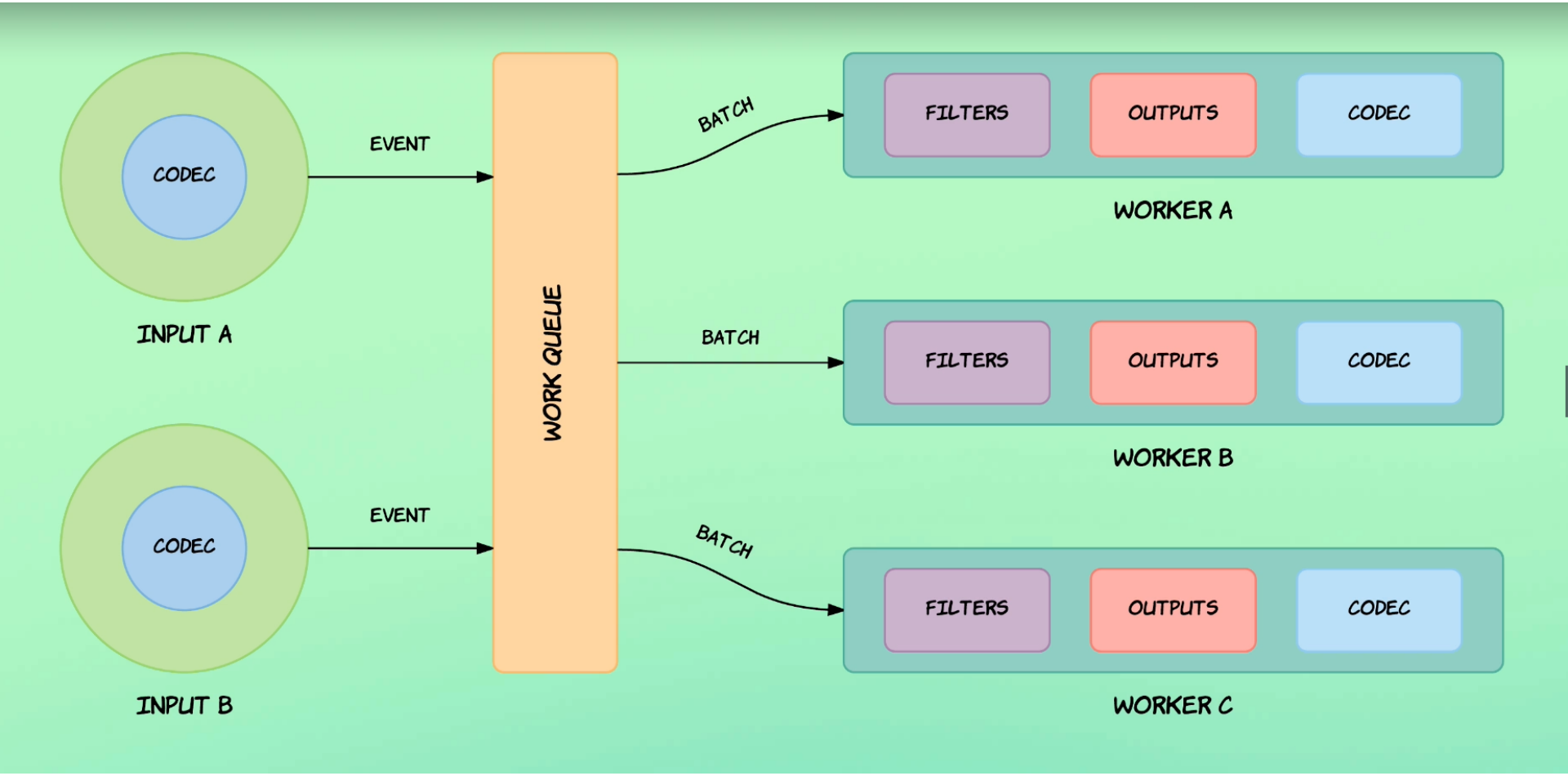
- It has the following key features:
- **Support for custom log formats**: Logs written by different applications often have particular formats specific to the application.
- Logstash helps parse and process custom formats on a large scale.
- It provides support to write your own filters for tokenization and also provides ready-to-use filters.

Logstash

- It has the following key features:

- **Plugin development**: Custom plugins can be developed and published, and there is a large variety of custom developed plugins already available.

Understanding Logstash Execution Model



Logstash

- Logstash enables any type of event to be enriched and transformed with a broad array of input, filter, and output plugins, with many native codecs further simplifying the ingestion process.
- Logstash provides insights by harnessing a greater volume and variety of data.
- Logstash can take input from various input mechanisms like files, Syslog, TCP/UDP, stdin, and many others.
- There is an extensive bouquet of filters that can be applied to the collected logs to transform the events.
- Logstash does not disappoint while outputting data because it supports multiple options like TCP/UDP, files, email, HTTP, Nagios, and many other network services.

Logstash

- Logstash has an extensible architecture and a developer-friendly plugin ecosystem.
- Logstash is the most popular event collection framework for consumption of data shipped from mobile devices to intelligent homes, connected vehicles, healthcare sensors, and many other industry-specific applications.
- It offers near real-time insights immediately at index or output time.
- Logstash offers many aggregations and mutations along with pattern matching, geo mapping, and dynamic lookup capabilities.
- Forwarding these logs from Logstash to Elasticsearch allows for performing a diverse range of mappings, aggregations, and searching.

Logstash

- Installing Logstash
- Logstash is written in Ruby and it's available as a tarball.
- Download from www.elastic.co/downloads/logstash.
- Version 7.7

Installing Logstash in Linux and MAC

tar –zxvf logstash.tsr.gz

```
Andys-MBP:Desktop Andy$ tar -zxvf logstash.tar.gz
```

Installing Logstash in Linux and MAC

```
ls: logs: No such file or directory
Andys-MBP:logstash Andy$ ls -l bin
total 80
-rwxr-xr-x@ 1 Andy  staff   377 Sep 25 22:29 cpdump
-rwxr-xr-x@ 1 Andy  staff   155 Sep 25 22:29 ingest-convert.sh
-rwxr-xr-x@ 1 Andy  staff  1654 Sep 25 22:29 logstash
-rwxr-xr-x@ 1 Andy  staff   448 Sep 25 22:29 logstash-plugin
-rw-r--r--@ 1 Andy  staff   228 Sep 25 22:29 logstash-plugin.bat
-rw-r--r--@ 1 Andy  staff  1449 Sep 25 22:29 logstash.bat
-rwxr-xr-x@ 1 Andy  staff  3677 Sep 25 22:29 logstash.lib.sh
-rwxr-xr-x@ 1 Andy  staff   840 Sep 25 22:29 ruby
-rw-r--r--@ 1 Andy  staff  1438 Sep 25 22:29 setup.bat
-rwxr-xr-x@ 1 Andy  staff  3530 Sep 25 22:29 system-install
Andys-MBP:logstash Andy$ bin/logstash -e "input { stdin { } } output { stdout { } }"
Sending Logstash's logs to /Users/Andy/Desktop/logstash/logs which is now configured via log4j2.properties
[2017-10-21T13:03:17,118][INFO ][logstash.modules.scaffold] Initializing module {:module_name=>"fb_apache", :directory=>"/Users/Andy/Desktop/
logstash/modules/fb_apache/configuration"}
[2017-10-21T13:03:17,125][INFO ][logstash.modules.scaffold] Initializing module {:module_name=>"netflow", :directory=>"/Users/Andy/Desktop/lo
gstash/modules/netflow/configuration"}
[2017-10-21T13:03:17,194][INFO ][logstash.setting.writabledirectory] Creating directory {:setting=>"path.queue", :path=>"/Users/Andy/Desktop/
logstash/data/queue"}
[2017-10-21T13:03:17,197][INFO ][logstash.setting.writabledirectory] Creating directory {:setting=>"path.dead_letter_queue", :path=>"/Users/A
ndy/Desktop/logstash/data/dead_letter_queue"}
[2017-10-21T13:03:17,248][WARN ][logstash.config.source.multilocal] Ignoring the 'pipelines.yml' file because modules or command line options
 are specified
[2017-10-21T13:03:17,273][INFO ][logstash.agent           ] No persistent UUID file found. Generating new UUID {:uuid=>"a752d529-fe0f-4d46-bd
ed-47251c31a640", :path=>"/Users/Andy/Desktop/logstash/data/uuid"}
[2017-10-21T13:03:17,604][INFO ][logstash.agent           ] Successfully started Logstash API endpoint {:port=>9600}
[2017-10-21T13:03:18,118][INFO ][logstash.pipeline        ] Starting pipeline {:pipeline_id=>"main", "pipeline.workers"=>8, "pipeline.batch.s
ize"=>125, "pipeline.batch.delay"=>5, "pipeline.max_inflight"=>1000, :thread=>"#<Thread:0x29d1caad@/Users/Andy/Desktop/logstash/logstash-core
/lib/logstash/pipeline.rb:290 run>"}
[2017-10-21T13:03:23,148][INFO ][logstash.pipeline        ] Pipeline started {"pipeline.id"=>"main"}
The stdin plugin is now waiting for input:
[2017-10-21T13:03:23,175][INFO ][logstash.agent           ] Pipelines running {:count=>1, :pipelines=>["main"]}
```

Logstash

GA RELEASE | PREVIEW RELEASE

Version:	6.5.3
Release date:	December 11, 2018
License:	Elastic License
Downloads:	TAR.GZ sha ZIP sha DEB sha RPM sha
Notes:	This default distribution is governed by the Elastic License, and includes the full set of free features. View the detailed release notes here. Not the version you're looking for? View past releases. The pure Apache 2.0 licensed distribution is available here. Java 8 is required for Logstash 6.x and 5.x.

Logstash

1. Download and unzip Logstash

   Logstash can also be installed from our package repositories using apt or yum. See *Repositories* in the Guide.

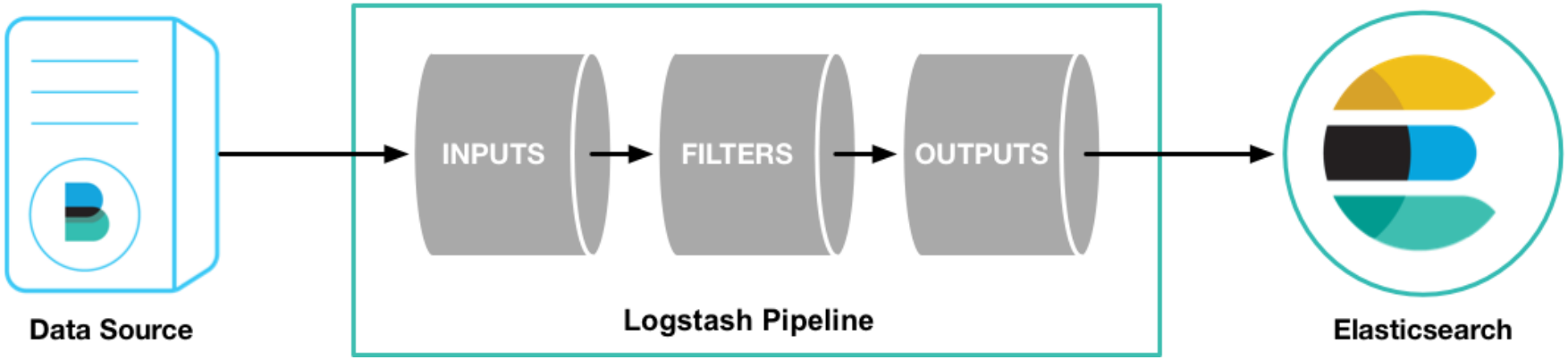
2. Prepare a logstash.conf config file

3. Run `bin/logstash -f logstash.conf`

4. Dive into the getting started guide and video.

Starting First Event in Logstash

- A Logstash pipeline has two required elements, input and output, and one optional element, filter.
- The input plugins consume data from a source, the filter plugins modify the data as you specify, and the output plugins write the data to a destination.



Logstash Running Logstash

logstash -e “input { stdin { } } output { stdout {} }”
Once Started type any message

```
D:\logstash-6.3.0\bin>logstash -e "input { stdin { } } output { stdout {} }"
Sending Logstash's logs to D:/logstash-6.3.0/logs which is now configured via log4j2.prope
rties
[2018-12-16T19:52:05,462][WARN ][logstash.config.source.multilocal] Ignoring the 'pipeline
s.yml' file because modules or command line options are specified
[2018-12-16T19:52:05,864][INFO ][logstash.runner          ] Starting Logstash {"logstash.v
ersion"=>"6.3.0"}
[2018-12-16T19:52:07,955][INFO ][logstash.pipeline        ] Starting pipeline {:pipeline_i
d=>"main", "pipeline.workers"=>8, "pipeline.batch.size"=>125, "pipeline.batch.delay"=>50}
[2018-12-16T19:52:08,070][INFO ][logstash.pipeline        ] Pipeline started successfully
{:pipeline_id=>"main", :thread=>"#<Thread:0x650529 run>"}
The stdin plugin is now waiting for input:
[2018-12-16T19:52:08,221][INFO ][logstash.agent           ] Pipelines running {:count=>1,
:running_pipelines=>[:main], :non_running_pipelines=>[]}
[2018-12-16T19:52:08,507][INFO ][logstash.agent           ] Successfully started Logstash
API endpoint {:port=>9600}
hello log
{
          "host" => "DESKTOP-55AGI0I",
       "message" => "hello log\r",
      "@version" => "1"
```

The -e flag enables quick testing of the configuration from the command line.

Logstash Running Logstash

logstash -e “input { stdin { } } output { stdout {} }”
Once Started type any message

From JDK 11 onwards
Logstash/config/jvm.options file
Add the following

```
-Djdk.io.File.enableADS=true
```

Logstash Sample conf file

```
input {
stdin { }
}
output {
stdout {
codec => rubydebug
}
}
```

Logstash Sample conf file

inputs: How events get into Logstash

- **filters**: How you can manipulate events in Logstash
- **outputs**: How you can output events from Logstash

- Each component block can have an associated plugin.
- In the example above, the input block has **stdin** plugin and the output block has **stdout** plugin.
- The stdout plugin has a codec with a value of **rubydebug**, which helps in outputting each event as a JSON hash.

Logstash Sample conf file

Type any message

```
D:\logstash-6.3.0\bin>logstash --verbose -f "D:\logstash configurations\sample.conf"
Sending Logstash's logs to D:/logstash-6.3.0/logs which is now configured via log4j2.prop
rties
[2018-12-16T20:14:13,989][WARN ][logstash.config.source.multilocal] Ignoring the 'pipelin
s.yml' file because modules or command line options are specified
[2018-12-16T20:14:14,542][INFO ][logstash.runner          ] Starting Logstash {"logstash.
ersion"=>"6.3.0"}
[2018-12-16T20:14:16,704][INFO ][logstash.pipeline        ] Starting pipeline {:pipeline_
d=>"main", "pipeline.workers"=>8, "pipeline.batch.size"=>125, "pipeline.batch.delay"=>50}
[2018-12-16T20:14:16,977][INFO ][logstash.pipeline        ] Pipeline started successfully
{:pipeline_id=>"main", :thread=>"#<Thread:0x1c6ae85 run>"}
The stdin plugin is now waiting for input:
[2018-12-16T20:14:17,044][INFO ][logstash.agent           ] Pipelines running {:count=>1,
:running_pipelines=>[:main], :non_running_pipelines=>[]}
[2018-12-16T20:14:17,374][INFO ][logstash.agent           ] Successfully started Logstash
API endpoint {:port=>9600}
Logstash started
{
      "@version" => "1",
       "message" => "Logstash started\r",
    "@timestamp" => 2018-12-16T14:45:24.855Z
```

Logstash Sample conf file

- The generated output contains the following components:

• "**message**": Includes the complete input message or the event line

• "**@timestamp**": Includes the timestamp of the time when the event was indexed; or if date filter is used, this value can also use one of the fields in the message to get a timestamp specific to the event

• "**host**": Represents the machine where this event was generated

Extending Logstash Functionality

- Logstash comes with several plugins, which can extend its functionality.
- These plugins come in the form of self-contained packages called gems and can be found at RubyGems.

Logstash Input Plugins

- **Beats**: This can be used to forward logs on servers to other machines for further processing. Being lightweight, it consumes minimal resources.
- **Date**: You can use this plugin to look for dates in fields. Thereafter, you can use that date as the logstash **@timestamp** for the event.
- **File**: This plugin constantly monitors files for any changes and pulls the new content as soon as it is appended.
- These new changes are then streamed as events.
- **Filter Plugins**: This plugin offers an optional facility where the original events can be modified and manipulated.

Sincedb_path

- When File{} input method reads a log file, the last byte processed is being saved and periodically copied out to the sincedb file.
- While you can set the file to be nul if you want, Logstash reads the file only during start up and uses the information from table in memory after.
- The problem is that the table in memory indexes position by inode, and is never pruned, even if it detects that a given file no longer exists.
- If you delete a file and then add a new one -- even if it has a different name -- it may well have the same inode number, and the File handler will think it is the same file.
- If the new file is larger, then the handler will only read from the previous max byte onwards and update the table.
- If the new file is smaller, then it seems to think the file was somehow truncated, and may start processing again from the default position.
- As a result, the only way to handle things is to set sincedb to be nul, and then restart logstash (causing the internal table to be lost) and then all the files matching the pattern will be re-read from the beginning - and this has problems as well, since some of the files may not be new.

Logstash Input Plugins

- **GEOIP**: This plugin fetches geographical location information from IP addresses. The logs are then enhanced with the location information.
- **Grok**: This plugin is the “heart and soul” of Logstash filters. It is quite popular for giving the proper form to unstructured data. You first define a search and then extract parts of the log line into structured fields.
- **Lumberjack**: This plugin utilizes the Lumberjack protocol to receive events. The Lumberjack protocol is not only secure, but is also reliable, has low latency offers, and needs lower resources.
- The use of the logstash-forwarder client makes it fast and lighter as compared to logstash.

Logstash Input Plugins

- **Multiline**: If you want to transform multiline messages from a single source into one logstash event, then go for this plugin.
- T**CP**: This is the best way to forward events coming over a TCP socket. Every event is treated as one line of text.

Logstash Codecs

- Codecs can be used to encode or decode output or input data. Some common codecs are the following:
- Default: Use the default “plain” codec for plain text with no delimitation between events.
- json: It encodes JSON events in inputs and decodes JSON messages in outputs.
- json_lines: Use this codec to receive and encode JSON events delimited by \n or to decode outputs with JSON messages delimited.
- rubydebug: This codec allows you to output Logstash events as data Ruby objects, thereby helping in debugging.

Logstash Output Plugins

- Logstash outputs are the end stage of the event pipeline. Before completing the event pipeline, you can use output plugins to forward the output to a particular destination.
- Some popular output plugins are the following:
- **Redis**: Redis is a very popular key-value in-memory data store and can be used as a buffer layer for the data pipeline. You can push the events to Redis by using the Redis plugin which utilizes **RPUSH**.
- **Kafka**: Kafka is a fast, scalable, and fault-tolerant commit log service. It can be used to provide the functionality of a **distributed messaging system**. You can use the Kafka plugin to write events to Kafka topic by leveraging the **Kafka Producer APIs**.

Logstash Output Plugins

- **Stdout**: This is plain vanilla simple output that prints to the stdout of the shell where logstash is running. It can be quite helpful for debugging plugin configurations

Logstash in and out(sample-json.conf)

- input {
- stdin {
- codec => json
- }
- }
- output {
- stdout {
- codec => rubydebug
- }
- }

Logstash in and out(sample-json.conf)

Type json data

Administrator: Command Prompt - logstash -f "D:\logstash configurations\sample-json.conf"

```
[2020-05-31T18:26:44,774][INFO ][logstash.inputs.stdin    ][main] Automatically switching from json to json_lines codec {:plugin=>"stdi
n"}
[2020-05-31T18:26:44,812][INFO ][logstash.javapipeline    ][main] Pipeline started {"pipeline.id"=>"main"}
The stdin plugin is now waiting for input:
[2020-05-31T18:26:44,861][INFO ][logstash.agent           ] Pipelines running {:count=>1, :running_pipelines=>[:main], :non_running_pip
elines=>[]}
[2020-05-31T18:26:45,070][INFO ][logstash.agent           ] Successfully started Logstash API endpoint {:port=>9600}
{"customerId":3423,"address":"chennai"}
D:/ELK/logstash-7.7.0/vendor/bundle/jruby/2.5.0/gems/awesome_print-1.7.0/lib/awesome_print/formatters/base_formatter.rb:31: warning: co
nstant ::Fixnum is deprecated
{
    "@timestamp" => 2020-05-31T12:58:21.879Z,
       "address" => "chennai",
      "@version" => "1",
    "customerId" => 3423,
          "host" => "DESKTOP-55AGI0I"
}
```

Logstash in and out(sample-json.conf)

Type json data

Administrator: Command Prompt - logstash -f "D:\logstash configurations\sample-json.conf"

```
The stdin plugin is now waiting for input:
[2020-05-31T18:26:44,861][INFO ][logstash.agent           ] Pipelines running {:count=>1, :running_pipelines=>[:main], :non_running_pip
elines=>[]}
[2020-05-31T18:26:45,070][INFO ][logstash.agent           ] Successfully started Logstash API endpoint {:port=>9600}
{"customerId":3423,"address":"chennai"}
D:/ELK/logstash-7.7.0/vendor/bundle/jruby/2.5.0/gems/awesome_print-1.7.0/lib/awesome_print/formatters/base_formatter.rb:31: warning: co
nstant ::Fixnum is deprecated
{
    "@timestamp" => 2020-05-31T12:58:21.879Z,
       "address" => "chennai",
      "@version" => "1",
    "customerId" => 3423,
          "host" => "DESKTOP-55AGI0I"
}
[{"customerId":1,"name":"Param"},{"customerId":2,"name":"Viki"}]
{
          "name" => "Param",
    "@timestamp" => 2020-05-31T13:02:14.434Z,
      "@version" => "1",
    "customerId" => 1,
          "host" => "DESKTOP-55AGI0I"
}
{
          "name" => "Viki",
    "@timestamp" => 2020-05-31T13:02:14.434Z,
      "@version" => "1",
    "customerId" => 2,
          "host" => "DESKTOP-55AGI0I"
}
```

Logstash json and file(sample-json-file.conf)

```
input {
stdin {
codec => json
}
}
output {
stdout {
codec => rubydebug
}
file{
 path=> "G:/Local disk/ELK/logs/output.txt"
}
}
```

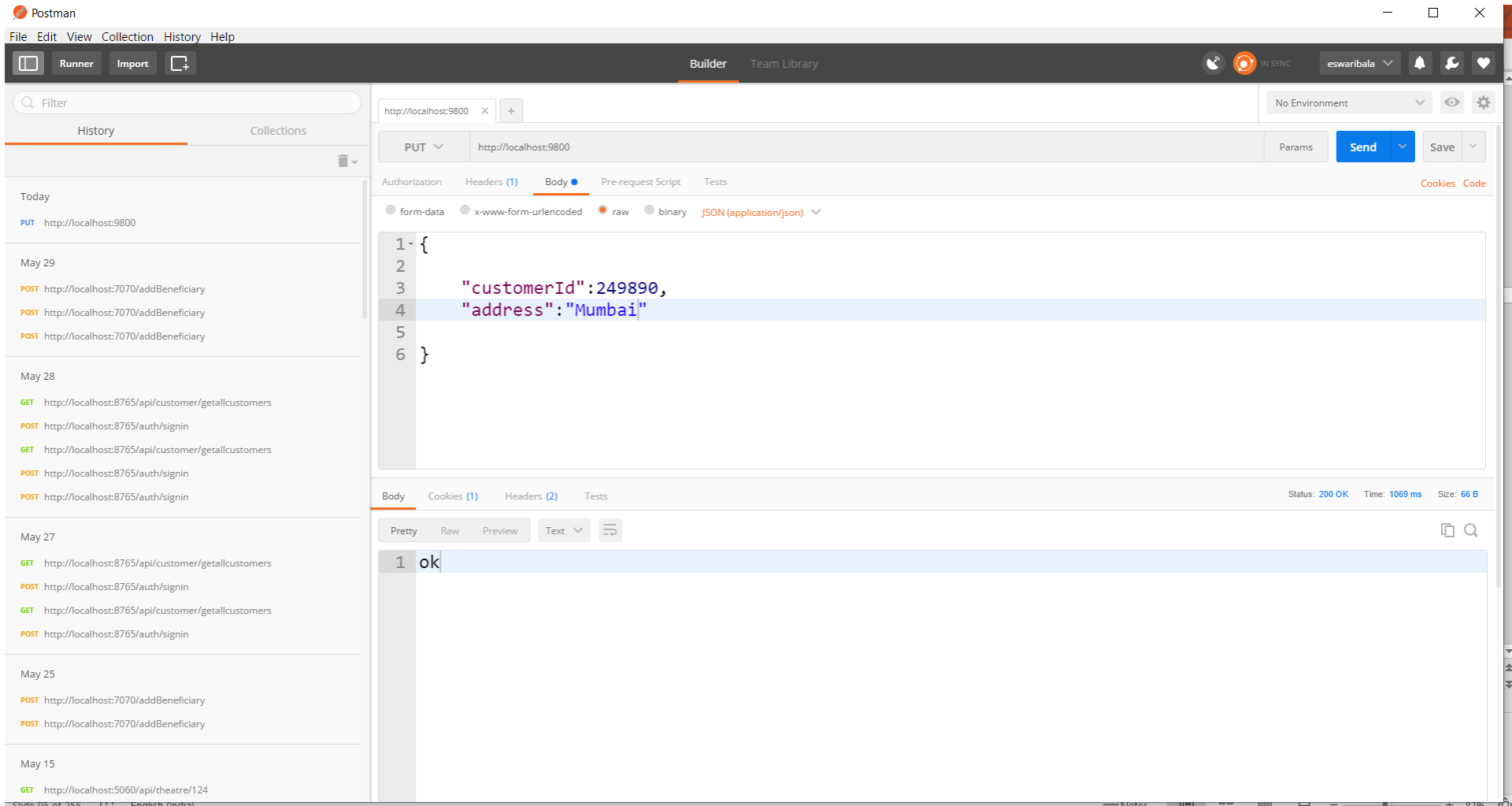
Logstash json and file(sample-json-file.conf)

```
The stdin plugin is now waiting for input:
[2020-05-31T18:38:34,274][INFO ][logstash.agent           ] Pipelines running {:count=>1, :running_pipelines=>[:main], :non_running_pip
elines=>[]}
[2020-05-31T18:38:34,480][INFO ][logstash.agent           ] Successfully started Logstash API endpoint {:port=>9600}
[{"customerId":32424,"address":"chennai"},{"customerId":32425,"address":"bangalore"}]
D:/ELK/logstash-7.7.0/vendor/bundle/jruby/2.5.0/gems/awesome_print-1.7.0/lib/awesome_print/formatters/base_formatter.rb:31: warning: co
nstant ::Fixnum is deprecated
{
       "address" => "chennai",
      "@version" => "1",
    "@timestamp" => 2020-05-31T13:09:33.824Z,
    "customerId" => 32424,
          "host" => "DESKTOP-55AGI0I"
}
{
       "address" => "bangalore",
      "@version" => "1",
    "@timestamp" => 2020-05-31T13:09:33.824Z,
    "customerId" => 32425,
          "host" => "DESKTOP-55AGI0I"
}
[2020-05-31T18:39:34,089][INFO ][logstash.outputs.file    ][main][bdab52ebe08e0883401236b297054468efb9b1041a81858c405d0769f10235c3] Ope
ning file {:path=>"G:/Local disk/ELK/logs/output.txt"}
```

Logstash json data from http

```
input {
stdin{
  codec => json
}
http{
  host => "127.0.0.1"
  port => "9800"
 }
}
output {
stdout {
codec => rubydebug
}
file{
 path=> "G:/Local disk/ELK/logs/output.txt"
}
}
```

Logstash json data from http



Logstash json data from http

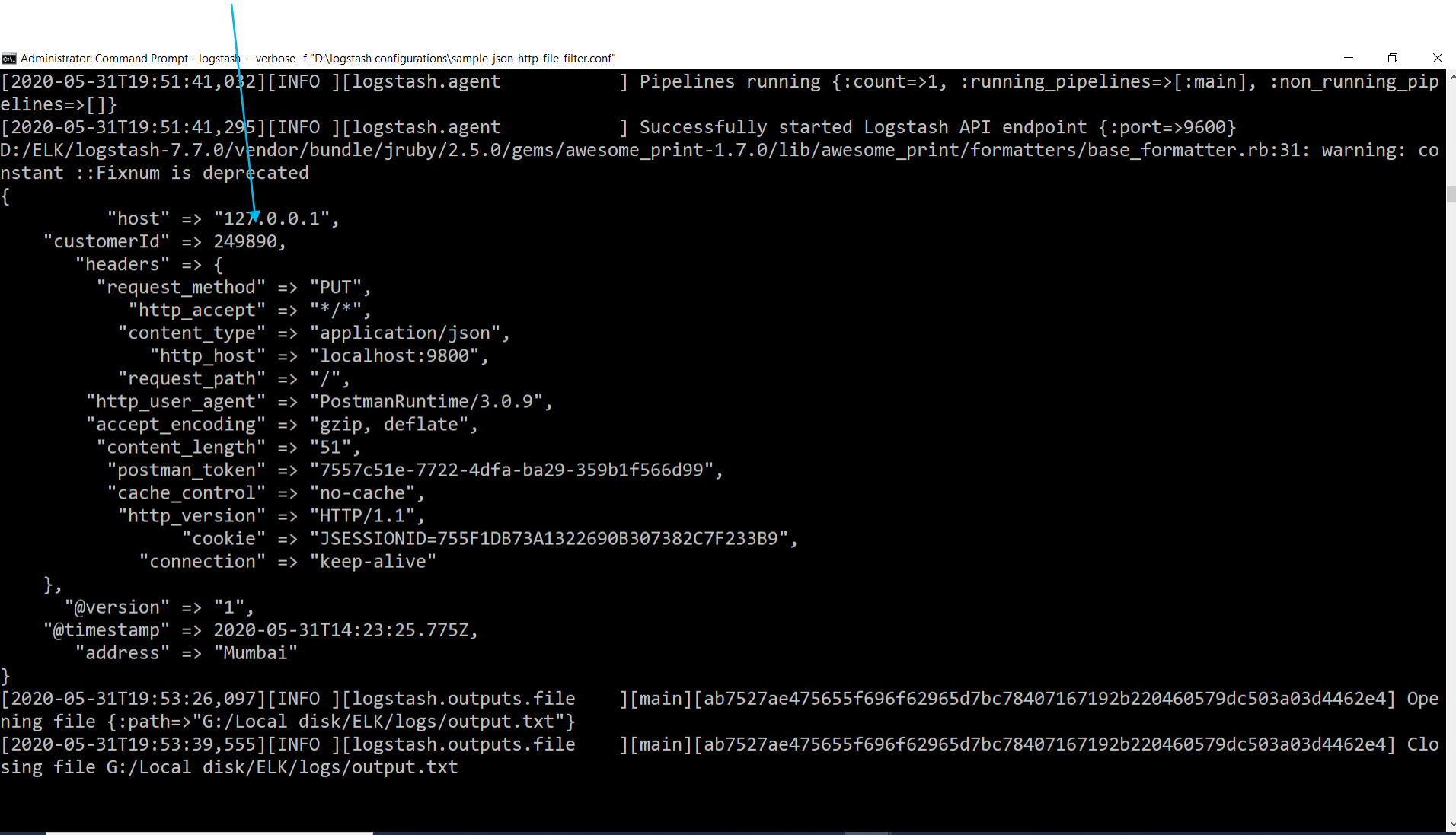
Administrator: Command Prompt - logstash --verbose -f "D:\logstash configurations\sample-json-http-file.conf"

```
[2020-05-31T19:28:07,411][INFO ][logstash.agent           ] Pipelines running {:count=>1, :running_pipelines=>[:main], :non_running_pip
elines=>[]}
[2020-05-31T19:28:07,675][INFO ][logstash.agent           ] Successfully started Logstash API endpoint {:port=>9600}

D:/ELK/logstash-7.7.0/vendor/bundle/jruby/2.5.0/gems/awesome_print-1.7.0/lib/awesome_print/formatters/base_formatter.rb:31: warning: co
nstant ::Fixnum is deprecated
{
          "host" => "127.0.0.1",
    "@timestamp" => 2020-05-31T14:00:09.013Z,
       "headers" => {
        "http_user_agent" => "PostmanRuntime/3.0.9",
           "request_path" => "/",
         "cache_control" => "no-cache",
       "accept_encoding" => "gzip, deflate",
        "content_length" => "49",
          "content_type" => "application/json",
         "postman_token" => "483c223b-71a2-4b2a-a0ab-a30037dfd2a9",
          "http_version" => "HTTP/1.1",
            "connection" => "keep-alive",
           "http_accept" => "*/*",
               "cookie" => "JSESSIONID=755F1DB73A1322690B307382C7F233B9",
            "http_host" => "localhost:9800",
       "request_method" => "PUT"
    },
    "customerId" => 249890,
      "@version" => "1",
       "address" => "Mumbai"
}
[2020-05-31T19:30:09,361][INFO ][logstash.outputs.file    ][main][8b8d52fd358de77e191e5827c3cc258e29a203ac56fd6febb112bf78b7670699] Ope
ning file {:path=>"G:/Local disk/ELK/logs/output.txt"}
[2020-05-31T19:30:26,511][INFO ][logstash.outputs.file    ][main][8b8d52fd358de77e191e5827c3cc258e29a203ac56fd6febb112bf78b7670699] Clo
sing file G:/Local disk/ELK/logs/output.txt
```

Logstash Filter Mutate (sample-json-http-file-filter)

```
input {
stdin{
  codec => json
}
http{
  host => "127.0.0.1"
  port => "9800"
 }
}
filter{
  mutate{
   convert => {"customerId" => "integer"}
        }
}
```

Logstash Filter Mutate (sample-json-http-file-filter)



http poller (

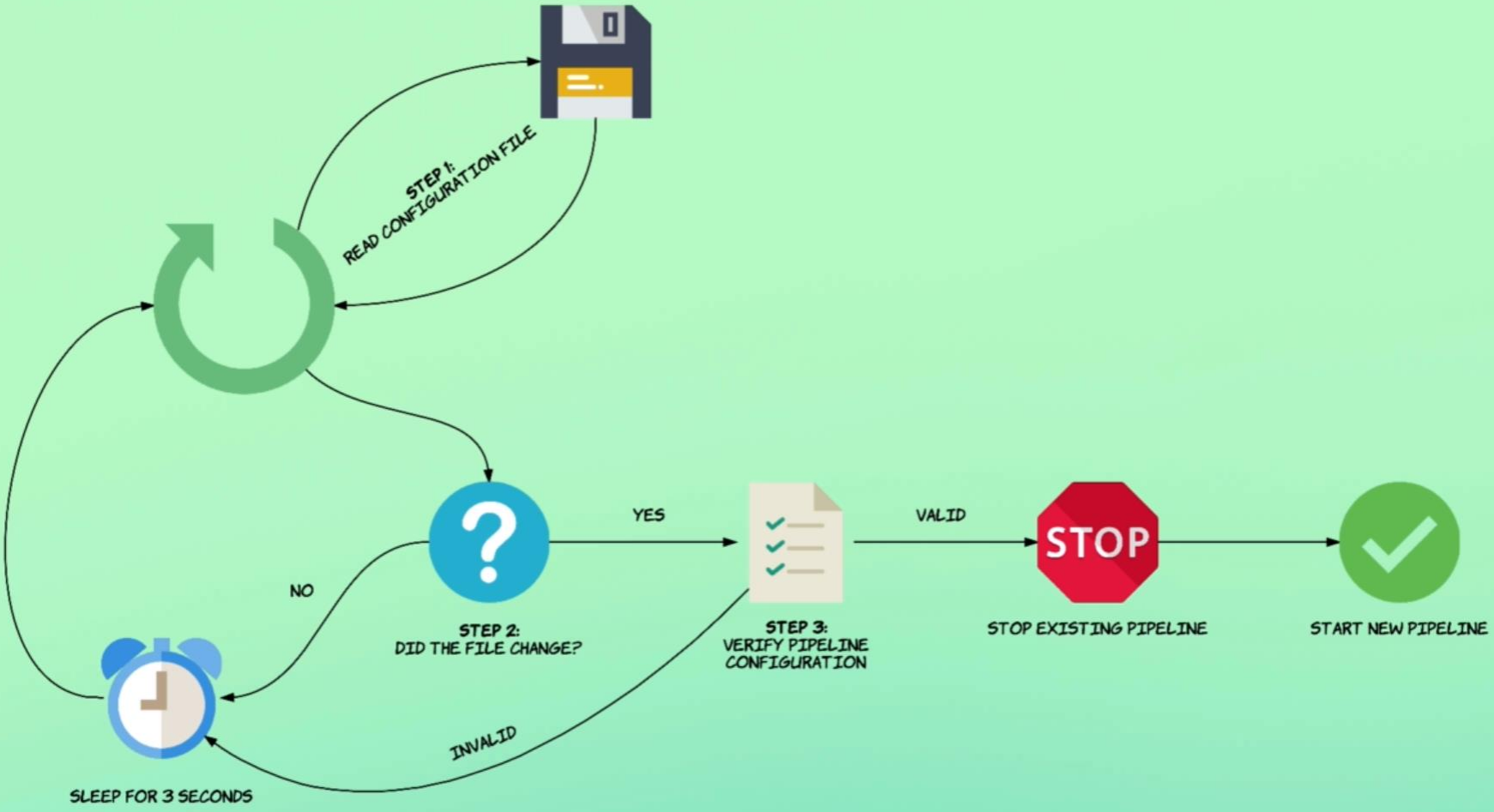
```
input {
 http_poller {
   urls => {
     myurl => "https://jsonplaceholder.typicode.com/users"
   }
codec => "json"
 }
}
```

- Note: “D:\logstash configurations\configurations\httppoller.conf”

Logstash Filter Options

Option	Purpose
`add_field`	Adds one or more fields to the event
`remove_field`	Removes one or more fields from the event
`add_tag`	Adds one or more tags to the event
`remove_tag`	Removes one or more tags from the event

Automatic Reload Configuration



Automatic Reload Configuration

Administrator: Command Prompt - logstash --verbose -f "D:\logstash configurations\sample-json-http-file-filter.conf" --config.reload.automatic

```
D:\ELK\logstash-7.7.0\bin>logstash --verbose -f "D:\logstash configurations\sample-json-http-file-filter.conf" --config.reload.automati
c
Java HotSpot(TM) 64-Bit Server VM warning: Option UseConcMarkSweepGC was deprecated in version 9.0 and will likely be removed in a futu
re release.
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by com.headius.backport9.modules.Modules (file:/D:/ELK/logstash-7.7.0/logstash-core/lib/jars/jruby-c
omplete-9.2.11.1.jar) to field java.io.Console.cs
WARNING: Please consider reporting this to the maintainers of com.headius.backport9.modules.Modules
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
Sending Logstash logs to D:/ELK/logstash-7.7.0/logs which is now configured via log4j2.properties
[2020-05-31T20:40:49,403][WARN ][logstash.config.source.multilocal] Ignoring the 'pipelines.yml' file because modules or command line o
ptions are specified
[2020-05-31T20:40:49,522][INFO ][logstash.runner          ] Starting Logstash {"logstash.version"=>"7.7.0"}
[2020-05-31T20:40:50,775][INFO ][org.reflections.Reflections] Reflections took 27 ms to scan 1 urls, producing 21 keys and 41 values
[2020-05-31T20:40:51,552][WARN ][org.logstash.instrument.metrics.gauge.LazyDelegatingGauge][main] A gauge metric of an unknown type (or
g.jruby.RubyArray) has been created for key: cluster_uuids. This may result in invalid serialization.  It is recommended to log an issu
e to the responsible developer/development team.
[2020-05-31T20:40:51,596][INFO ][logstash.javapipeline    ][main] Starting pipeline {:pipeline_id=>"main", "pipeline.workers"=>8, "pipe
line.batch.size"=>125, "pipeline.batch.delay"=>50, "pipeline.max_inflight"=>1000, "pipeline.sources"=>["D:/logstash configurations/samp
le-json-http-file-filter.conf"], :thread=>"#<Thread:0x218fc601 run>"}
[2020-05-31T20:40:52,352][INFO ][logstash.inputs.stdin    ][main] Automatically switching from json to json_lines codec {:plugin=>"stdi
n"}
[2020-05-31T20:40:52,490][INFO ][logstash.javapipeline    ][main] Pipeline started {"pipeline.id"=>"main"}
[2020-05-31T20:40:52,500][INFO ][logstash.inputs.http     ][main][10c28e34ef9a5ef631f41ff1238aadcb7893e64dd44166ba18eae4c3f56be608] Sta
rting http input listener {:address=>"127.0.0.1:9800", :ssl=>"false"}
The stdin plugin is now waiting for input:
[2020-05-31T20:40:52,560][INFO ][logstash.agent           ] Pipelines running {:count=>1, :running_pipelines=>[:main], :non_running_pip
elines=>[]}
[2020-05-31T20:40:52,804][INFO ][logstash.agent           ] Successfully started Logstash API endpoint {:port=>9600}
D:/ELK/logstash-7.7.0/vendor/bundle/jruby/2.5.0/gems/awesome_print-1.7.0/lib/awesome_print/formatters/base_formatter.rb:31: warning: co
nstant ::Fixnum is deprecated
```

Storing logs with Elastic

- logstash -e “input { stdin { } } output { elasticsearch { hosts => localhost } }”

- After starting type some message

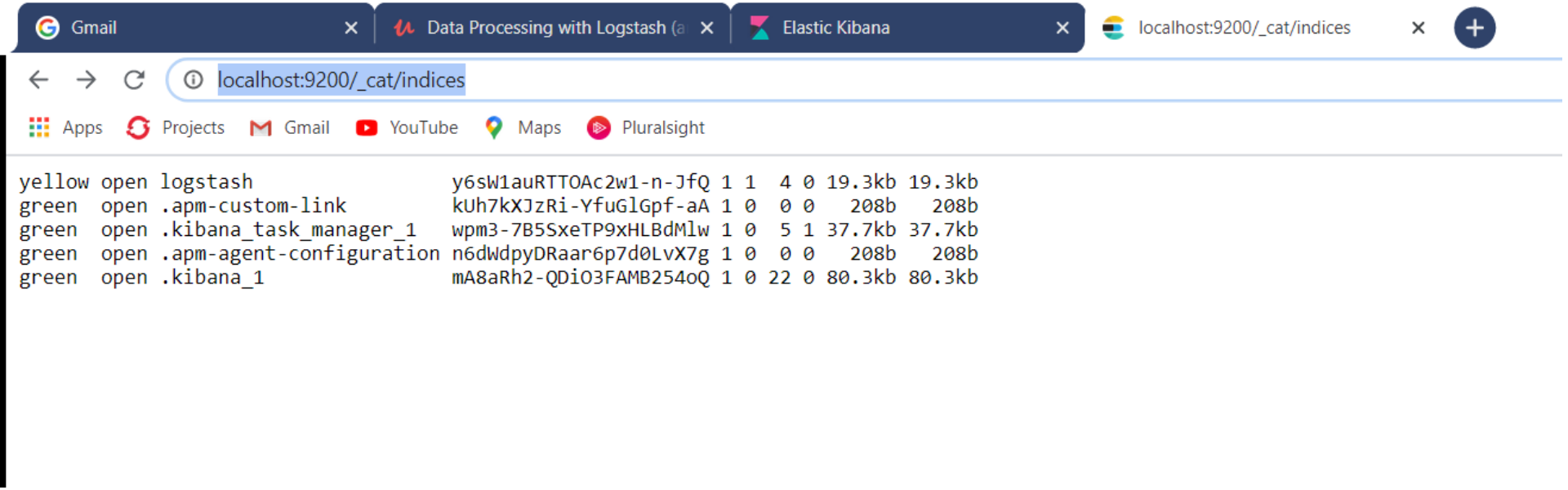
Storing logs with Elastic

```
refresh_interval"=>"5s"}, "mappings"=>{"_default_"=>{"dynamic_templates"=>[{"message_field
"=>{"path_match"=>"message", "match_mapping_type"=>"string", "mapping"=>{"type"=>"text", "
norms"=>false}}}, {"string_fields"=>{"match"=>"*", "match_mapping_type"=>"string", "mappin
g"=>{"type"=>"text", "norms"=>false, "fields"=>{"keyword"=>{"type"=>"keyword", "ignore_abo
ve"=>256}}}}}], "properties"=>{"@timestamp"=>{"type"=>"date"}, "@version"=>{"type"=>"keywo
rd"}, "geoip"=>{"dynamic"=>true, "properties"=>{"ip"=>{"type"=>"ip"}, "location"=>{"type"=
>"geo_point"}, "latitude"=>{"type"=>"half_float"}, "longitude"=>{"type"=>"half_float"}}}}}
}}}
[2018-12-16T21:19:35,813][INFO ][logstash.outputs.elasticsearch] Installing elasticsearch
template to _template/logstash
[2018-12-16T21:19:38,135][INFO ][logstash.outputs.elasticsearch] New Elasticsearch output
{:class=>"LogStash::Outputs::ElasticSearch", :hosts=>["//localhost"]}
[2018-12-16T21:19:38,255][INFO ][logstash.pipeline        ] Pipeline started successfully
{:pipeline_id=>"main", :thread=>"#<Thread:0xf28c60 run>"}
The stdin plugin is now waiting for input:
[2018-12-16T21:19:38,348][INFO ][logstash.agent           ] Pipelines running {:count=>1,
:running_pipelines=>[:main], :non_running_pipelines=>[]}
[2018-12-16T21:19:38,630][INFO ][logstash.agent           ] Successfully started Logstash
API endpoint {:port=>9600}
Hi
How are you
```

Storing logs with Elastic

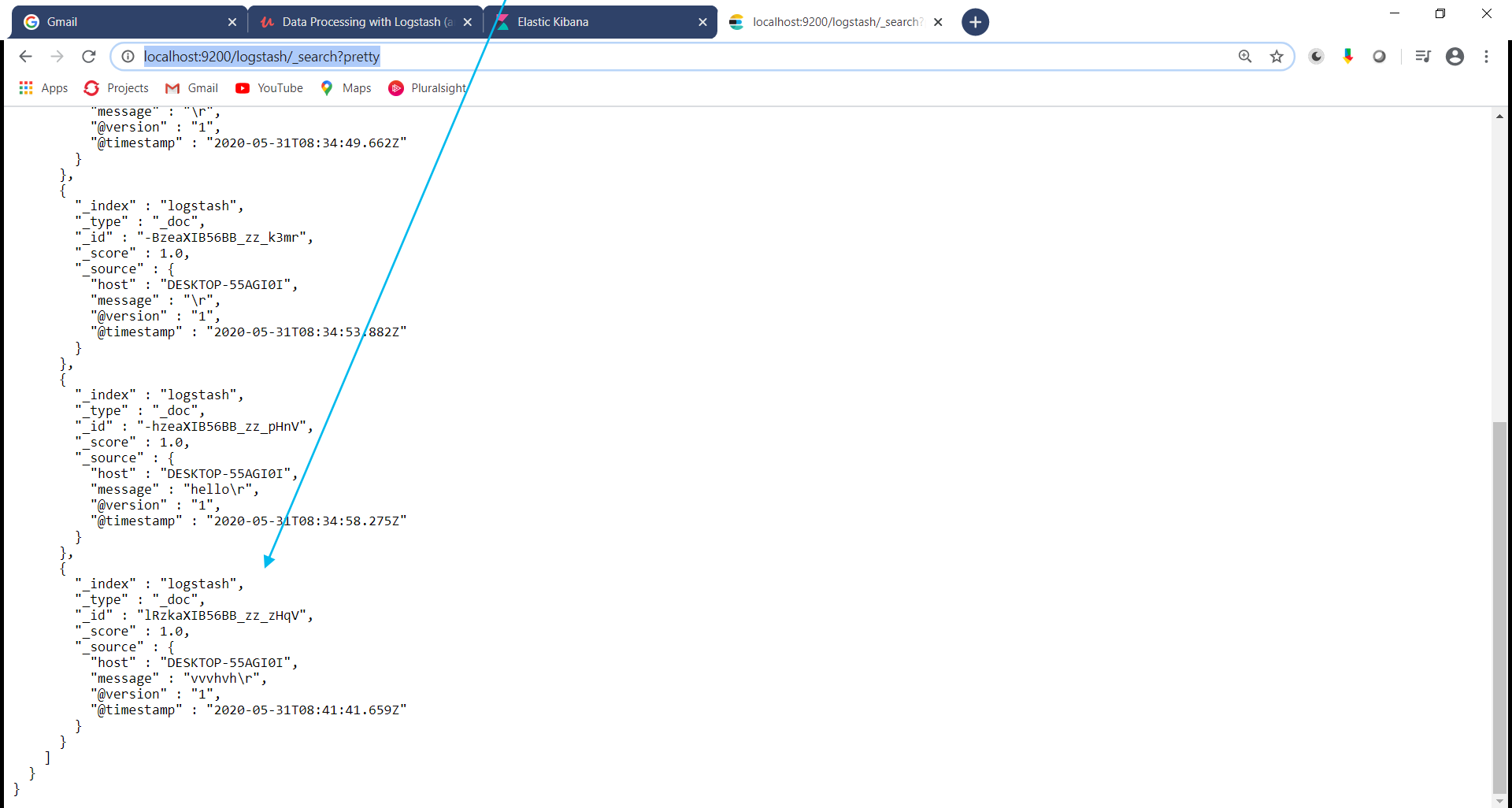
http://localhost:9200/_cat/indices

http://localhost:9200/_cat/indices?v&health=yellow



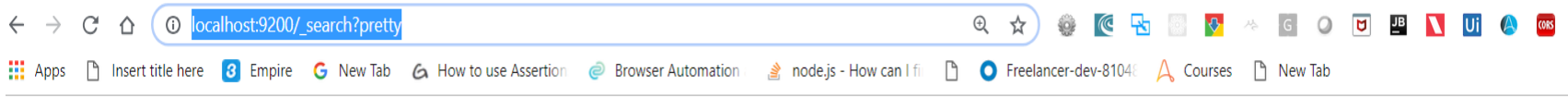
Storing logs with Elastic

http://localhost:9200/logstash/_search?pretty



Storing logs with Elastic

http://localhost:9200/_search?pretty




```
{
  "took" : 58,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 1,
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "logstash-2018.12.16",
        "_type" : "doc",
        "_id" : "ZZG2t2cBNWJlpFD3QMCG",
        "_score" : 1.0,
        "_source" : {
          "message" : "Hi\r",
          "@timestamp" : "2018-12-16T15:50:31.020Z",
          "host" : "DESKTOP-55AGI0I",
          "@version" : "1"
        }
      }
    ]
  }
}
```

Delete All from Elastic

- curl -X DELETE http://localhost:9200/_all


```
{
  "took" : 0,
  "timed_out" : false,
  "_shards" : {
    "total" : 0,
    "successful" : 0,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 0,
    "max_score" : 0.0,
    "hits" : [ ]
  }
}
```

Capturing Tomcat logs

```
D:\logstash-6.3.0\bin>logstash --verbose -f "D:\logstash configurations\pipeline.conf"
Sending Logstash's logs to D:/logstash-6.3.0/logs which is now configured via log4j2.prope
rties
[2018-12-16T21:52:36,947][WARN ][logstash.config.source.multilocal] Ignoring the 'pipeline
s.yml' file because modules or command line options are specified
[2018-12-16T21:52:37,388][INFO ][logstash.runner          ] Starting Logstash {"logstash.v
ersion"=>"6.3.0"}
[2018-12-16T21:52:38,921][INFO ][logstash.pipeline        ] Starting pipeline {:pipeline_i
d=>"main", "pipeline.workers"=>8, "pipeline.batch.size"=>125, "pipeline.batch.delay"=>50}
[2018-12-16T21:52:39,472][INFO ][logstash.outputs.elasticsearch] Elasticsearch pool URLs u
pdated {:changes=>{:removed=>[], :added=>[http://localhost:9200/]}}
[2018-12-16T21:52:39,481][INFO ][logstash.outputs.elasticsearch] Running health check to s
ee if an Elasticsearch connection is working {:healthcheck_url=>http://localhost:9200/, :p
ath=>"/"}
[2018-12-16T21:52:39,669][WARN ][logstash.outputs.elasticsearch] Restored connection to ES
 instance {:url=>"http://localhost:9200/"}
[2018-12-16T21:52:39,729][INFO ][logstash.outputs.elasticsearch] ES Output version determi
ned {:es_version=>6}
[2018-12-16T21:52:39,733][WARN ][logstash.outputs.elasticsearch] Detected a 6.x and above
cluster: the `type` event field won't be used to determine the document _type {:es_version
=>6}
```

Capturing Tomcat logs


```
{
  "took" : 8,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 46,
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "logstash-2018.12.16",
        "_type" : "doc",
        "_id" : "MjHdt2cBiVdjzqqkDY82",
        "_score" : 1.0,
        "_source" : {
          "@version" : "1",
          "@timestamp" : "2018-12-16T16:32:56.520Z",
          "message" : "How are you\r",
          "host" : "DESKTOP-55AGI0I"
        }
      },
      {
        "_index" : "logstash-2018.12.16",
        "_type" : "doc",
        "_id" : "ODHet2cBiVdjzqqkBY_S",
        "_score" : 1.0,
        "_source" : {
          "@version" : "1",
          "@timestamp" : "2018-12-16T16:34:00.032Z",
          "message" : "28-Apr-2018 04:41:26.094 INFO [main] org.apache.catalina.startup.VersionLoggerListener.log Java Home:            C:\\Program Files\\Java\\jre1.8.0_161\r",
          "path" : "C:/Program Files/Apache Software Foundation/Tomcat 9.0/logs/catalina.2018-04-28.log",
          "host" : "DESKTOP-55AGI0I"
        }
      },
      {
        "_index" : "logstash-2018.12.16",
        "_type" : "doc",
        "_id" : "OTHet2cBiVdjzqqkBY_S",
```

Shipping, Filtering, and Parsing Events with Logstash

- Sample Dataset
- This dataset can be downloaded from the following location:
- http://cdiac.ornl.gov/ftp/us_recordtemps/sta424/tmax_serial/CA_6719tmax.txt

Shipping, Filtering, and Parsing Events with Logstash

- The sample data has the following fields:
- Station Number (i.e., 046719: two-digit state code, followed by
- four-digit station code)
- Day of Year (1-365)
- Year
- Month
- Day of Month
- Tmax (Maximum Temperature)

Shipping, Filtering, and Parsing Events with Logstash

Station Number	Day of Year	Year	Month	Day of Month	Maximum Temperature
046719	1	1986	1	1	62
046719	2	1986	1	2	63
046719	3	1986	1	3	65
046719	4	1986	1	4	66
046719	5	1986	1	5	66
046719	6	1986	1	6	73
046719	7	1986	1	7	74
046719	8	1986	1	8	72
046719	9	1986	1	9	76
046719	10	1986	1	10	82

Shipping, Filtering, and Parsing Events with Logstash

- 1,1986-01-01,62
- 2,1986-01-02,63
- 3,1986-01-03,65
- 4,1986-01-04,66
- 5,1986-01-05,66
- 6,1986-01-06,73
- 7,1986-01-07,74
- 8,1986-01-08,72
- 9,1986-01-09,76

Logstash Configuration

- Before proceeding with log analysis, Logstash has to be configured to accept input from a particular source and in a particular format.
- In order to read, parse, and filter different types of data, Logstash enables you to specify different types of inputs, outputs, and filters.
- This is facilitated by a diverse set of plugins. In order to read data from a file, the **file** plugin can be used.

Logstash Configuration

- Each line in the source file is treated as a separate event and streamed by the file input plugin.
- In running systems, typically the log files rotate. The file input plugin has the ability to detect file rotation and handle it accordingly.
- This it does by maintaining the last read location.
- New data is automatically detected if the correct configuration is done..

Logstash Configuration

- input {
- file {
- path => #String (path of the files) (required)
- start_position => #String (optional, default "end")
- tags => #array (optional)
- type => #string (optional)
- }
- }

Logstash Configuration

- **path**: It is the only mandatory field in the file input plugin and is used to specify the path of the file from where input events have to be received and processed.
- **start_position**: Logstash can start reading from any point in the source file and this point is specified by **“start_position”**.
- It can take the value of **“beginning”** or **“end”**. In order to read live streams, specify the default value of “end”.
- Only if you want to read any historical data do you need to specify a value as “beginning”.

Logstash Configuration

- **tags**: This field helps in filtering and processing events. Any number of filter strings can be specified as an array for this purpose.
- **type**: In order to mark a specific type of events, you can categorize a specific type of events by using this field. Type is added to a document that is indexed in Elasticsearch.
- It can be later viewed in Kibana under the **_type** field. For example, you can assign type as **“critical”** or **“warning”**.

Logstash Configuration

- The configuration varies according to the plugin type. Often there are cases where a plugin may require the value to be of a certain type such as string or array. The following value types are supported:
- **Array**: If you want to specify multiple values, then use the array type. Specifying the same setting multiple times appends to the array.
- For Example:
- path => ["/var/log/*.log", "/var/log/postgresql/*.log"]
- path => "/var/log/apache/*.log"
- This example specifies **path** to be an array with an element for each of the strings.

Logstash Configuration

- **Boolean**: The value of a Boolean type can be either **true** or **false**. Take care that the true and false keywords are not within quotes.
- For example,
- ssl_enable => false
- **Bytes**: This field is a string field representing a valid unit of bytes.
- It provides a convenient way to use specific sizes in plugin options. It support both **base-1000 SI** (k M G T P E Z Y) and **base-1024 Binary** (Ki Mi Gi Ti Pi Ei Zi Yi) units. Not only are these fields case-insensitive but they also accept spaces between value and unit. If no unit is specified, the integer string stands for the number of bytes.

Logstash Configuration

- **Codec**: This field represents name of the Logstash codec being used for input or output.
- **Input** codecs facilitate decoding data before actual processing.
- **Output** codecs facilitate encoding data before outputting it. By using an input or output codec, you eliminate the need for using a separate filter. See the following for an example:
- codec => "plain"
- **Hash**: A collection of **key value pairs** in the form “field1” => “value1”. The comma separator is used to separate multiple key value entries. See the following for an example:
- match => {
- "key1" => "value1"
- "key2" => "value2"
- ...
- }

Logstash Configuration

- **Codec**: This field represents name of the Logstash codec being used for input or output.
- **Input** codecs facilitate decoding data before actual processing.
- **Output** codecs facilitate encoding data before outputting it. By using an input or output codec, you eliminate the need for using a separate filter. See the following for an example:
- codec => "plain"
- **Hash**: A collection of **key value pairs** in the form “field1” => “value1”. The comma separator is used to separate multiple key value entries. See the following for an example:
- match => {
- "key1" => "value1"
- "key2" => "value2"
- ...
- }

Logstash Configuration

- **Number**: It represents valid **numeric** values (floating point or integer). For example, num_descriptor => 25
- • **Password**: Represents a string that will be neither **logged** nor **printed**. For example,
- admin_password => "password"
- • **Path**: Used to specify a valid system path. For example, log_path => /var/opt/log
- **String**: String values are single character sequences enclosed in **quotes** (double or single). You need to use the **backslash** to escape literal quotes if they are of the same kind as the string delimiter. You need to escape both double quotes within a double-quoted string and single quotes within a single-quoted string. For example,
- name => "Good Bye"
- name => 'It\'s a hot afternoon'
- name => "I like \"red\" shirts

Logstash Configuration

- Comments
- You can put comments in the configuration file in the same way as is done in Perl, Ruby, and Python. You can start the comment with a # **character**, and it can be in any position in a line, like so:
- # Comment from start of the line
- input { # Comment at the end of line
- # ...
- }

Logstash Configuration

- Configuring for Events
- The Logstash event pipeline consists of three stages: **inputs**, **filters**, and **outputs**. Events are generated by inputs and modified by filters. Events are shipped by outputs.
- The event properties are referred to as **fields** by Logstash. For example, an HTTP request has an HTTP verb like GET or PUT. Event-specific configuration can be done in Logstash.
- Since events are generated by inputs, the event-specific configuration applies only after the input phase. The event-specific configuration works only within filter and output blocks.

Logstash Configuration

- Field References
- It can be more intuitive to refer to a field by name, and this is exactly what the Logstash field reference syntax achieves. You can access a field by **[fieldname]**.
- For **top-level fields**, just **omit** the [] and simply use fieldname. For **nested fields**, specify the full path to that field: **[top-level-field][nested field]**. For example, the following event has two nested fields and one top level field:

{

"network": {

"ip": ["192.168.1.21"],

"timeout": 20,

},

"path": "/var/log/syslog"

}

- To reference the **timeout** field, you specify **[network][timeout]**. To reference a toplevel
- field such as **path**, just specify the field name.

Logstash Configuration

- sprintf Format
- You can use field reference format in sprintf format also. This way you can refer to field values from within other strings.
- For example, the statsd output can increment field values like timeout.

```
output {
statsd {
increment => "syslog.%{[response][status]}"
}
```

Logstash Configuration

- Conditionals
- In certain scenarios, you may want to filter or output an event under certain conditions.
- This is where you can use a conditional. Logstash conditionals are pretty similar to their programming language counterpart. Logstash supports if, else, and else statements, which can be nested also.
- The syntax looks like the following:

```
if EXPRESSION {

...

} else if EXPRESSION {

...

} else {

...

}
```

Logstash Configuration

- Comparison uses Boolean logic to arrive at the correct path. The following
- comparison operators can be used:
- The **equality** operators are ==, !=, <, >, <=, >=.
- The **regexp** operators of =~, !~ check a pattern on the right-hand side against a string value on the left-hand side.
- The **inclusion** operators are in and not in.

You can use the following binary operators:

- **and, or, xor, nand**
- You can use the following unary operator:
- **!**

Logstash Configuration

- There are lots of permutations and combinations possible with expressions. They can include other expressions or group few expressions using parenthesis (…).
- In the following expression, a conditional check is used to take an action, which in this case is to **drop** events that contain DEBUG or INFO level log information:

```
filter {
#Rest of the processing
if [type] == "linux-syslog" and [messagetype] in ["DEBUG", "INFO"]
{
drop {}
}
}
```

Logstash Configuration

- Multiple expressions can be combined in a single condition.

```
output {
# Send production errors to stdout
if [loglevel] == "ERROR" and [type] == "apache-error" {
stdout { codec => rubydebug }
}
}
```

Logstash Configuration

- Metadata
- From Logstash 1.5 onwards, you can specify metadata with events. It is a neat way to extend and build event fields with field references and sprintf formatting.
- The metadata information is specified by using @**metadata** field.
- A common use for metadata tag is to handle logs of different applications running on the same machine.
- Each application emits its own logs in a separate log file.
- The local Logstash reads all the messages, processes, and forwards ahead to the central Logstash server or Elasticsearch server.
- It can be challenging to ensure that the correct filters and output run on the logs.

Logstash Configuration

- Adding the “tags” field to the input and checking for it in filters and output is a common practice. This requires some discipline in removing the tag in the output.
- More often than not, the tag stays in the output and leads to unexpected results.
- This can now be avoided by intelligently adding metadata to events.
- In fact, a metadata tag can be used to form an independent Logstash pipeline for each application running on the same system without the need of running multiple instances of Logstash.

Logstash Configuration

- The following example shows how to use metadata tags for RabbitMQ logs.
- On reading logs from a RabbitMQ topic and processing them, each type of log is dumped into its own RabbitMQ topic (based on the type field of the event).
- Refer Separate-logs.conf

Logstash Configuration

- Filtering Events
- Before proceeding with log analysis, Logstash has to be configured to accept input from a particular source and in a particular format.
- In order to read, parse, and filter different types of data, Logstash lets you specify different types of inputs, outputs, and filters.
- This is facilitated by a diverse set of plugins. In order to read data from a file, the **file** plugin can be used.

Logstash Configuration

- After configuring the input file, the appropriate filter needs to be applied on the input so that only useful fields are picked and analyzed.
- For this purpose, a **filter** plugin can be used to perform intermediate processing on the input event. This filter can be applied on selective fields based on conditions.
- Since your input file is in a CSV format, it is best to use the **csv** filter. On receiving the input event, the csv filter parses it and stores its individual fields. Besides the comma, it can parse data with other separators also.

Logstash Configuration

- Generally, the csv filter looks like following:
- filter {
- csv {
- columns => #Array of column names.
- separator => #String ; default -","
- }
- }
- Optionally, the attribute columns can be used to specify the name of fields in an input csv file. The default nomenclature would be column1, column2, and so on.
- The separator attribute specifies the character to be used to separate the different columns in the file.

Logstash Configuration

- For your example, let's use the following csv filter:

```
filter {
csv {
columns =>
["day_of_year","date_of_record","max_temp"]
separator => ","
}
}
```

Logstash Configuration

- There is a specific **date filter** in Logstash and it looks as following:

```
filter {
date {
match => # array (optional), default: []
target => # string (optional), default: "@timestamp"
timezone => # string (optional)
}
}
```

Logstash Configuration

- The match attribute is associated with an array in the format [field, formats].
- It is followed by a set of time formats which can be applied to the field.
- If the input events have multiple formats, the following code can be used:

match => ["date_field", "MMM dd YYY HH:mm:ss",

"ISO8601", "MMddYYYY", "MMM d YYY HH:mm:ss"]

Logstash Configuration

- Based on the input event date format, the date filter would be the following:

```
date{
match => ["date_of_record", "yyyy-MM-dd"]
target => "@timestamp"
}
```

Logstash Configuration

- For conversion of fields to a specific data type, the **mutate** filter can be used.
- This filter performs general mutations such as modification of data types, renaming, replacing fields, and removing fields.
- It can also perform other advanced functions like merging two fields, performing uppercase and lowercase conversion, split and strip fields, and so on.
- Generally, a **mutate** filter looks like following:

Logstash Configuration

```
filter {
mutate {
convert => # hash of field and data type (optional)
join => # hash of fields to be joined (optional)
lowercase => # array of fields to be converted (optional)
merge => # hash of fields to be merged (optional)
rename => # hash of original and rename field (optional)
replace => # hash of fields to replaced with (optional)
split => # hash of fields to be split (optional):
```

Logstash Configuration

- strip => # array of fields (optional)
- uppercase => # array of fields (optional)
- }
- }
- The mutate filter in your case looks like the following:
- mutate {
- convert => ["max_temp","integer"]
- }
- The convert functionality is being used to convert max_temp (maximum temperature) to an integer.

Logstash Configuration

- Shipping Events
- After transforming data into a CSV format, configuring Logstash to accept data from a CSV file, and process it based on the specified data type, you are all set to ship the events.
- In your example, Logstash will fetch the data from the CSV file and ship it to **Elasticsearch**, where the different fields can be indexed.
- This will facilitate the visualization of data using the **Kibana** interface.
- The **output** plugin of Logstash can be used to get output in a form acceptable by Elasticsearch.

Logstash Configuration

- output

```
{
elasticsearch {
action => # string (optional), default: "index"
hosts => # array
document_id => # string (optional), default: nil
index => # string (optional), default: "logstash-%{+YYYY.MM.dd}"
path => # string (optional), default: "/”
timeout => # number
}
}
```

Logstash Configuration

- **action**: The action to take on incoming documents. The default action is “**index**” which can be changed to “delete”. For indexing a document, use the “index” value; for deleting a document, use the “delete” value.
- **hosts**: IP address or hostname(s) of the node(s) where Elasticsearch is running. If multiple hosts are specified, requests will be load balanced.
- For example, a single host can be specified as “127.0.0.1” and multiple hosts can be specified as [“127.0.0.1:9200”, “127.0.0.2:9200”].

Logstash Configuration

- **document_id**: Document id of the index; useful to delete or overwrite the existing entries.
- **index**: The index name where incoming events have to be written.
- The default action is to index based on each day and name it as “logstash-%{+YYYY.MM.dd}”. The timestamp value is based on the filter criteria (event capture time or event raising time).
- **path**: HTTP path at which Elasticsearch is accessible.
- **timeout**: The timeout value for network requests and requests send to Elasticsearch.

Logstash Configuration

- logstash –f “D:\logstash configurations/tmax.conf”


```
{
  "took" : 7,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 9,
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "tmax-data",
        "_type" : "doc",
        "_id" : "L1vDvGcB1da7Pa9bD8Dw",
        "_score" : 1.0,
        "_source" : {
          "message" : "5,05/01/1986,66\r",
          "path" : "D:/logstash configurations/tmax.csv",
          "host" : "DESKTOP-55AGI0I",
          "day_of_year" : "5",
          "max_temp" : 66,
          "date_of_record" : "05/01/1986",
          "@timestamp" : "1986-01-04T18:30:00.000Z",
          "@version" : "1"
        }
      },
      {
        "_index" : "tmax-data",
        "_type" : "doc",
        "_id" : "LFvDvGcB1da7Pa9bD8Dv",
        "_score" : 1.0,
        "_source" : {
          "message" : "1,01/01/1986,62\r",
          "path" : "D:/logstash configurations/tmax.csv",
          "host" : "DESKTOP-55AGI0I",
          "day_of_year" : "1",
          "max_temp" : 62,
          "date_of_record" : "01/01/1986",
          "@timestamp" : "1985-12-31T18:30:00.000Z",
          "@version" : "1"
        }
```

Logstash Configuration

- To list all indices
- http://localhost:9200/_cat/indices?v
- To search by index
- curl http://localhost:9200/tmax-data/_search?pretty*
- To search by type
- http://localhost:9200/_all/doc/_search?pretty
- To delete by index
- curl -XDELETE “http://localhost:9200/.monitoring-es-6-2018.12.17”

Logstash Configuration Ruby filter

```
filter {
csv {
      separator => ","
#Date,Open,High,Low,Close,Volume (BTC),Volume (Currency),Weighted Price
      columns => ["Date","Open","High","Low","Close","Volume (BTC)", "Volume (Currency)" ,"Weighted Price"]
  }

  ruby {
    code => "
      wanted_fields = ['High', 'Low']
      event.to_hash.keys.each { |k|
        event.remove(k) unless wanted_fields.include? k
      }
    "
  }
}
```

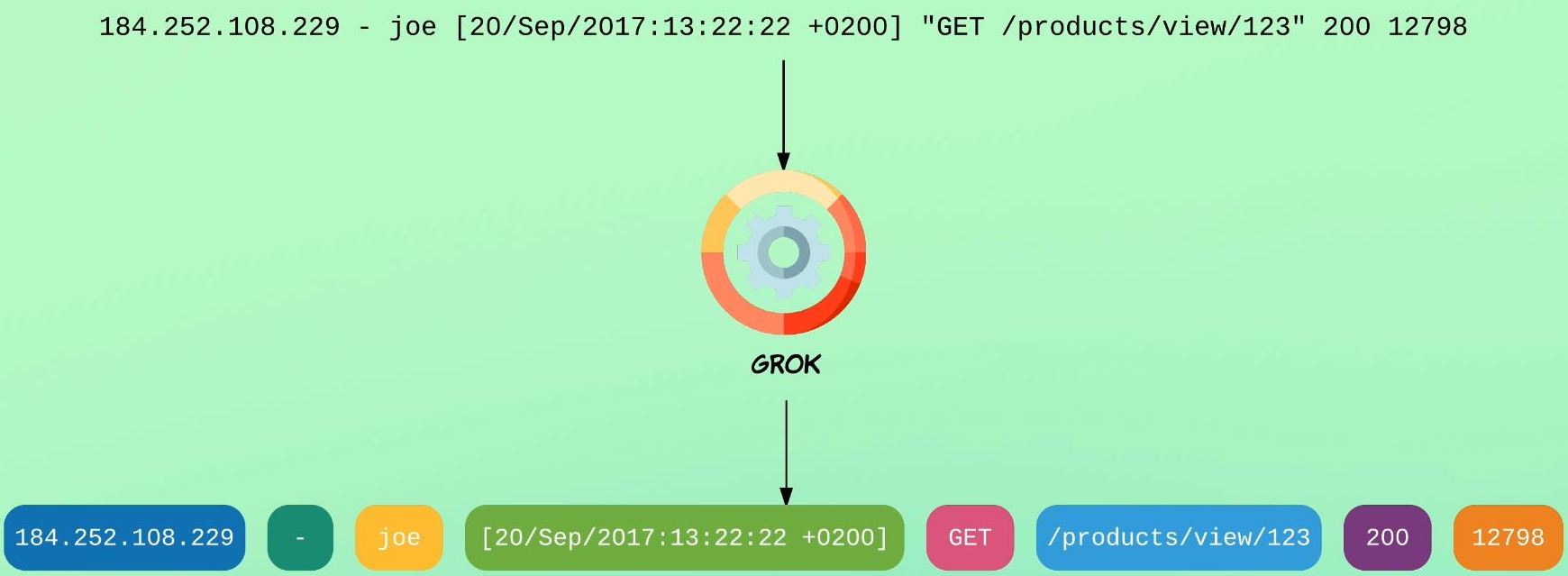
Logstash Configuration Ruby filter


```
{
  "took" : 27,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 1321,
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "bitcoinruby-prices",
        "_type" : "doc",
        "_id" : "XCEJwmcBk_dKDbM7Zl8H",
        "_score" : 1.0,
        "_source" : {
          "High" : "973",
          "Low" : "912.02001"
        }
      },
      {
        "_index" : "bitcoinruby-prices",
        "_type" : "doc",
        "_id" : "WCEJwmcBk_dKDbM7ZmAL",
        "_score" : 1.0,
        "_source" : {
          "High" : "973.2",
          "Low" : "915.15"
```

Logstash Configuration grok

- Grok is a great way to parse unstructured log data into something structured and queryable.
- GROK (Graphical Representation of Knowledge)
- This tool is perfect for syslog logs, apache and other webserver logs, mysql logs, and in general, any log format that is generally written for humans and not computer consumption.
- Logstash ships with about 120 patterns by default.
- Grokpatterns [click here]

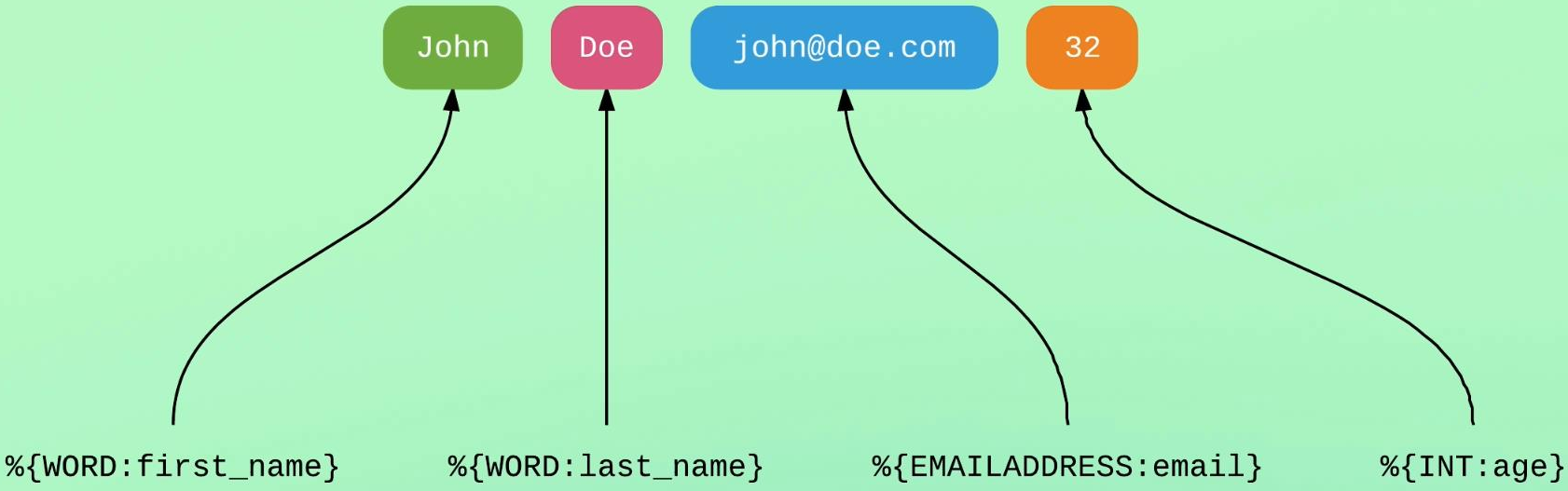
Parsing Requests using grok



Parsing Requests using grok

```
%{SYNTAX:SEMANTIC}
```

Parsing Requests using grok



Parsing file request using grok

- https://github.com/logstash-plugins/logstash-patterns-core/blob/master/patterns/grok-patterns

- https://www.elastic.co/guide/en/logstash/current/plugins-filters-grok.html

- Refer ngnixgrok.conf

Date grok(dategrok.conf)

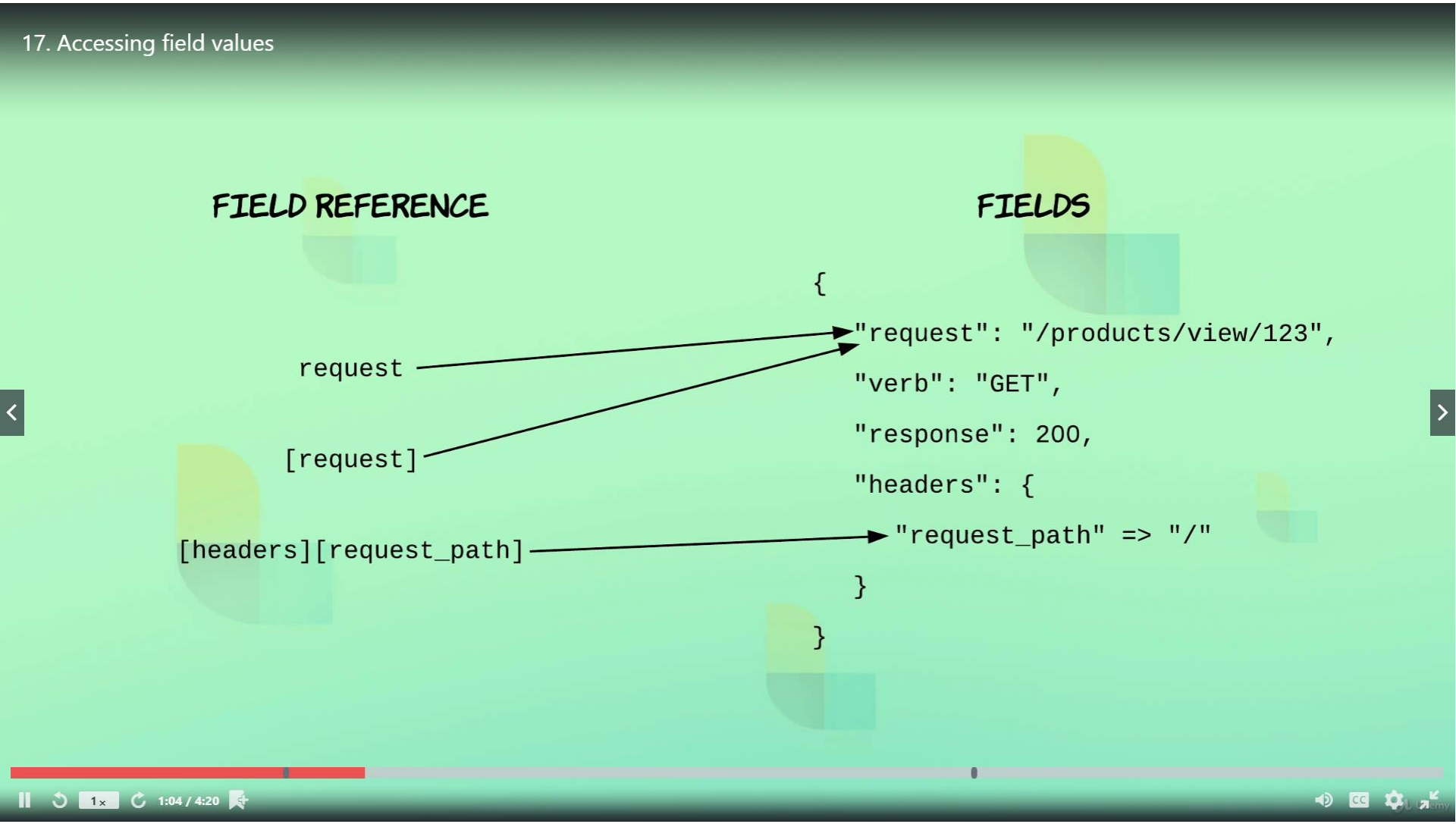
- 127.0.0.1 - - [11/Dec/2013:00:01:45 -0800] "GET /xampp/status.php HTTP/1.1" 200 3891 "http://cadenza/xampp/navi.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/20100101 Firefox/25.0"

Date grok(dategrok.conf)

Administrator: Command Prompt - logstash --verbose -f "D:\logstash configurations\date_grok.conf" --config.reload.automatic

```
          "bytes" => "3891",
      "timestamp" => "11/Dec/2013:00:01:45 -0800",
       "@version" => "1",
     "@timestamp" => 2013-12-11T08:01:45.000Z,
       "clientip" => "127.0.0.1",
        "message" => "127.0.0.1 - - [11/Dec/2013:00:01:45 -0800] \"GET /xampp/status.php HTTP/1.1\" 200 3891 \"http://cadenza/xampp/navi.php\" \"Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/20100101 Firefox/25.0\"\r",
        "request" => "/xampp/status.php",
       "referrer" => "\"http://cadenza/xampp/navi.php\"",
           "verb" => "GET",
           "host" => "DESKTOP-55AGI0I"
}
127.0.0.1 - - [11/Dec/2013:00:01:45 -0800] "GET /xampp/status.php HTTP/1.1" 200 3891 "http://cadenza/xampp/navi.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/20100101 Firefox/25.0"
{
          "ident" => "-",
           "auth" => "-",
          "agent" => "\"Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/20100101 Firefox/25.0\"",
       "response" => "200",
    "httpversion" => "1.1",
          "bytes" => "3891",
      "timestamp" => "11/Dec/2013:00:01:45 -0800",
       "@version" => "1",
     "@timestamp" => 2013-12-11T08:01:45.000Z,
       "clientip" => "127.0.0.1",
        "message" => "127.0.0.1 - - [11/Dec/2013:00:01:45 -0800] \"GET /xampp/status.php HTTP/1.1\" 200 3891 \"http://cadenza/xampp/navi.php\" \"Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/20100101 Firefox/25.0\"\r",
        "request" => "/xampp/status.php",
       "referrer" => "\"http://cadenza/xampp/navi.php\"",
           "verb" => "GET",
           "host" => "DESKTOP-55AGI0I"
}
```

Access Field Value



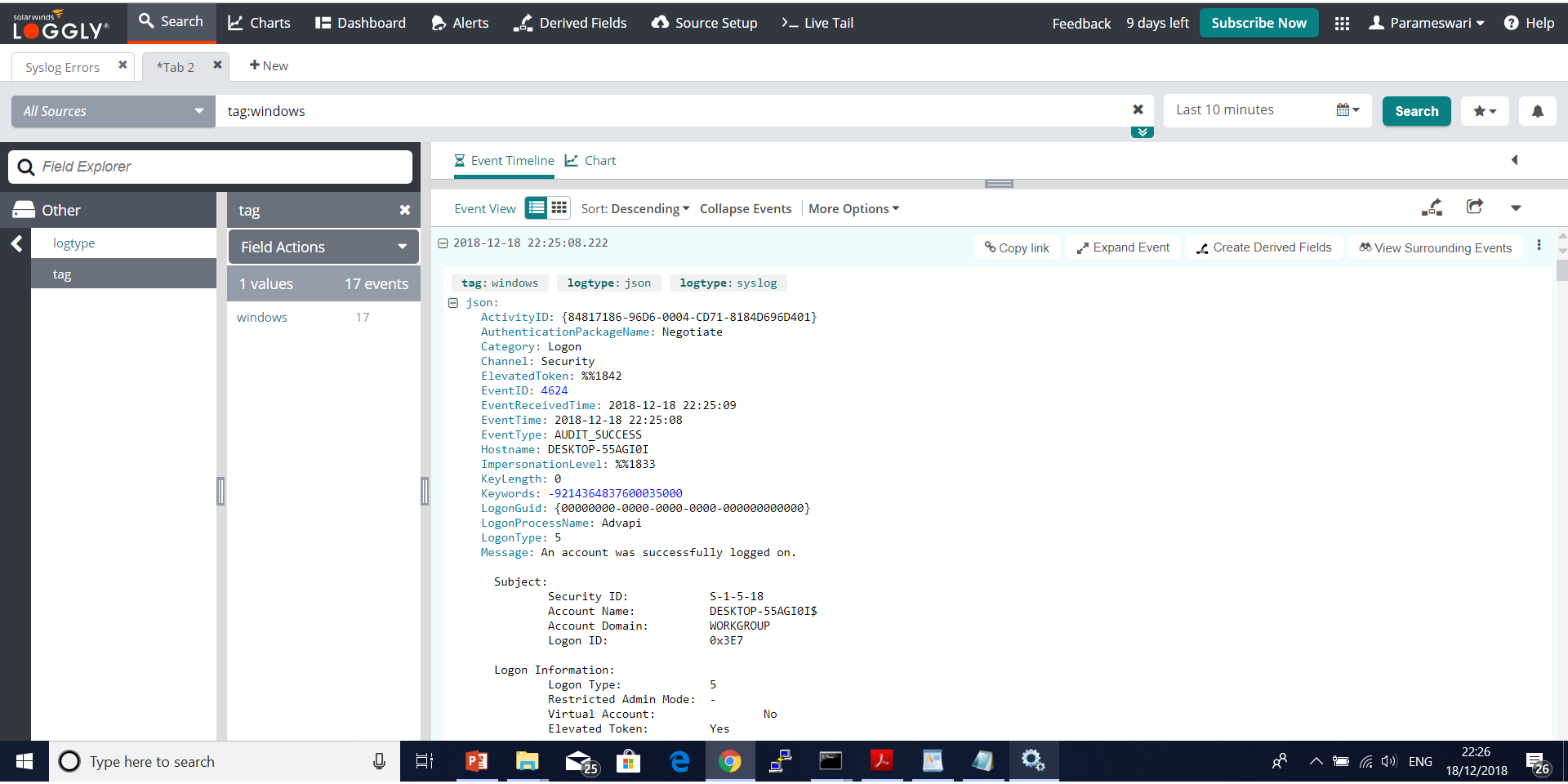
Syslog with grok

- Download nxlog
- https://nxlog.co/products/nxlog-community-edition/download
- Install nxlog msi
- **Copy the Configuration**
- **Replace** the above variables:
- CUSTOMER_TOKEN: Replace with your own customer token
- ROOT and ROOT_STRING: If you are in 32-bit Windows, uncomment the top root path on lines 8 and 9 to use the 32-bit program files folder then comment the two below.

Syslog with grok

- Verify
- Verify it shows up in Loggly by doing a search for the windows tag over the past hour.
- tag:windows

Syslog with grok



Geoip logstash

Administrator: RabbitMQ Command Prompt (sbin dir) - logstash -f "D:\logstash configurations\geotest.conf"

```
{
    "@timestamp" => 2018-12-18T19:17:29.840Z,
          "geoip" => {
            "region_name" => "Victoria",
                     "ip" => "1.1.1.1",
         "continent_code" => "OC",
              "city_name" => "Research",
          "country_code3" => "AU",
            "postal_code" => "3095",
              "longitude" => 145.15,
               "location" => {
              "lat" => -37.7333,
              "lon" => 145.15
         },
          "country_code2" => "AU",
               "timezone" => "Australia/Melbourne",
           "country_name" => "Australia",
              "latitude" => -37.7333,
           "region_code" => "VIC"
    },
          "host" => "DESKTOP-55AGI0I",
      "@version" => "1",
```

Extending Logstash

- Plugin Management
- Logstash has an extensive collection of plugins (**inputs, filters, outputs, codecs**).
- These plugins are developed by Elasticsearch but also get significant contributions from the community.
- The biggest forte of Logstash is the ease of availability of plugins and the flexibility of adding new ones to provide more features.
- In fact, about **200** plugins are available for you to choose from and work with.

Extending Logstash

- Plugins in Logstash are not part of the core package.
- Ruby is used to develop Logstash plugins.
- The Ruby programming language comes with a package manager called RubyGems.
- This package manger specifies a common format for distributing programs and libraries built in Ruby.
- It manages the installation and distribution of gems.
- RubyGems is used to make Logstash plugins available as separate self-contained packages.
- RubyGems provides the support required for the release of plugin updates independently from Logstash releases.

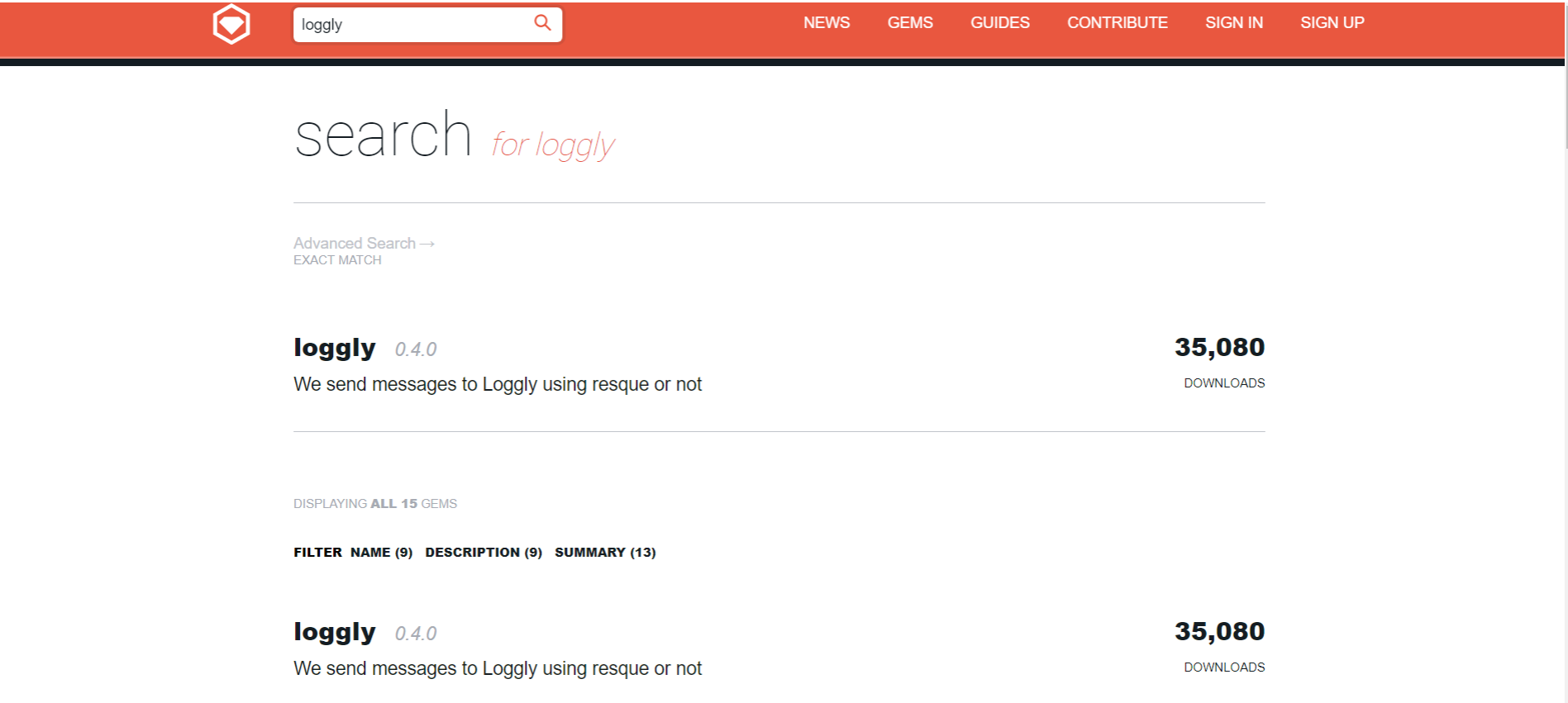
Extending Logstash

- The key advantages of moving plugins away from the Logstash core are as follows:
- The Logstash release can be independent of plugin updates.
- Developers can release new features and bug fixes for plugins without being tied to the Logstash release plan.
- It offers easy-to-describe external dependencies.
- It means a leaner core Logstash distribution package.

Extending Logstash

- Logstash plugins, both core and community ones, can be downloaded from
- https://rubygems.org/.
- The GitHub repository is another place where you can access all of the Logstash plugins:
- https://github.com/logstash-plugins.

Extending Logstash



Plugin Installation

- Let's say you want to install the logstash-output-mongodb plugin.
- You can issue the following command from the Logstash installation folder:
- **logstash-plugin install logstash-output-mongodb**
- This command will install the **logstash-output-mongodb** plugin to the Logstash installation.
- If you want to install a particular version, you can specify the –version parameter.
- logstash-plugin list

Plugin Installation

- Updating a Plugin
- Whenever a new version of a plugin is released, you may want to upgrade the existing installation of that plugin.
- This can be done by using the following command:
- logstash-plugin update logstash-output-mongodb
- This command will update the logstash-output-mongodb plugin to its latest version.

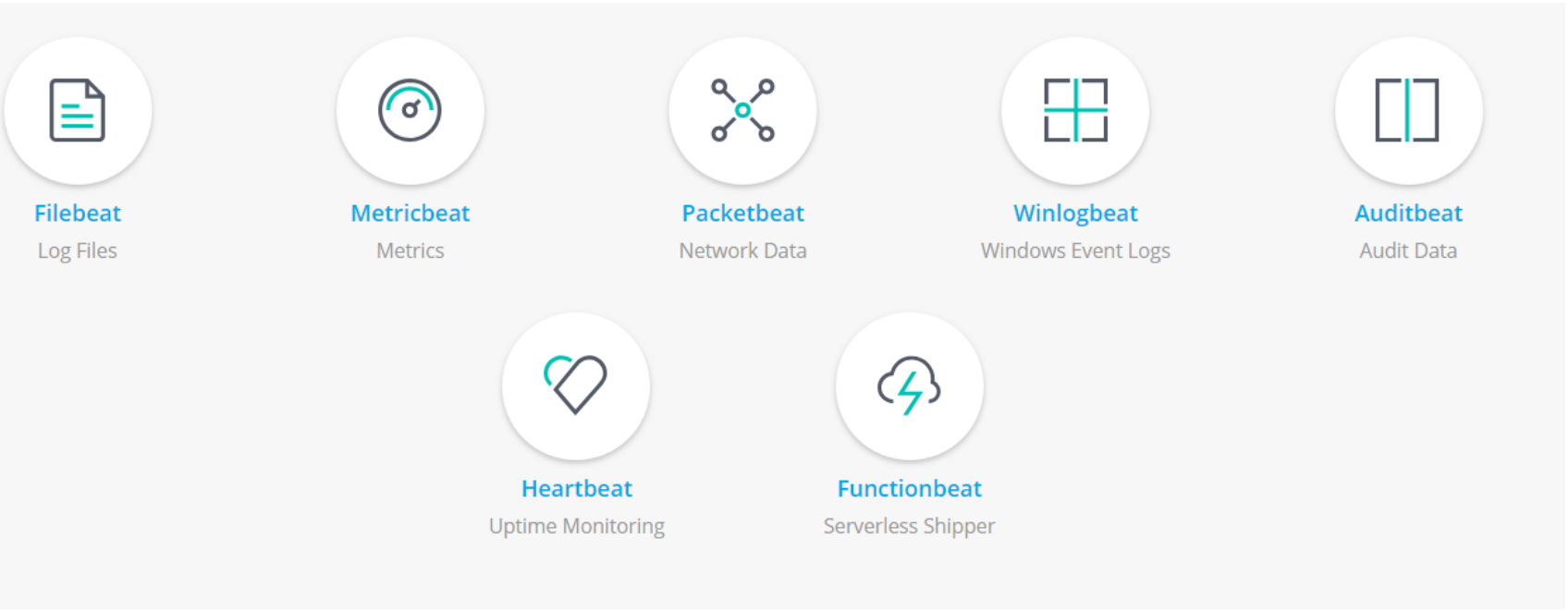
Plugin Installation

- Uninstallation
- If you no longer wish to use a particular plugin, you can uninstall it in the following manner:
- logstash-plugin uninstall logstash-output-mongodb
- This command will **uninstall** the logstash-event plugin from the Logstash installation..

Lightweight Data Shippers

- Beats is the platform for single-purpose data shippers.
- They send data from hundreds or thousands of machines and systems to Logstash or Elasticsearch.

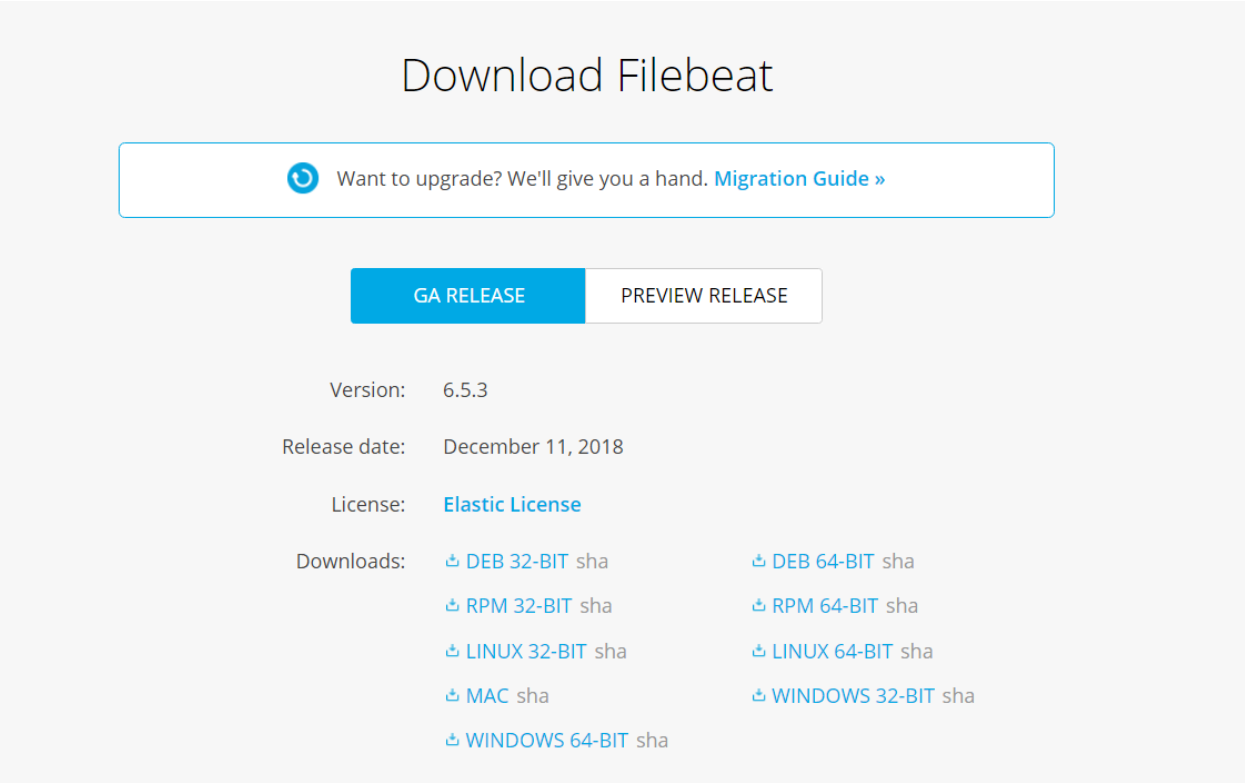
Lightweight Data Shippers



Lightweight Data Shippers

- Filebeat: collects and ships log files.
- Metricbeat: collects metrics from your systems and services.
- Packetbeat: collects and analyzes network data.
- Winlogbeat: collects Windows event logs.
- Auditbeat: collects Linux audit framework data and monitors file integrity.
- Heartbeat: monitors services for their availability with active probing.

Install FileBeat



Install FileBeat

Installation Steps

1. Download and unzip Filebeat

   Filebeat can also be installed from our package repositories using apt or yum. See *Repositories* in the Guide.

2. Edit the filebeat.yml configuration file

3. Start the daemon by running `sudo ./filebeat -e -c filebeat.yml`

4. Dive into the getting started guide and video.

File Beat Configuration

- Open the Filebeat configuration file:
- filebeat/filebeat.yml
- Filebeat supports numerous outputs, but you'll usually only send events directly to Elasticsearch or to Logstash for additional processing.
- Here we'll use Logstash to perform additional processing on the data collected by Filebeat.
- Filebeat will not need to send any data directly to Elasticsearch, so let's disable that output.
- To do so, find the output.elasticsearch section and comment out the following lines by preceding them with a #:

File Beat Configuration

- Then, configure the output.logstash section.
- Uncomment the lines output.logstash: and hosts: ["localhost:5044"] by removing the #.
- This will configure Filebeat to connect to Logstash on your Elastic Stack server at port 5044, the port for which we specified a Logstash input earlier:

File Beat Configuration

- The functionality of Filebeat can be extended with Filebeat modules.
- Here we will use the system module, which collects and parses logs created by the system logging service of common Linux distributions.
- Let's enable it:
- filebeat modules enable system
- You can see a list of enabled and disabled modules by running:
- filebeat modules list

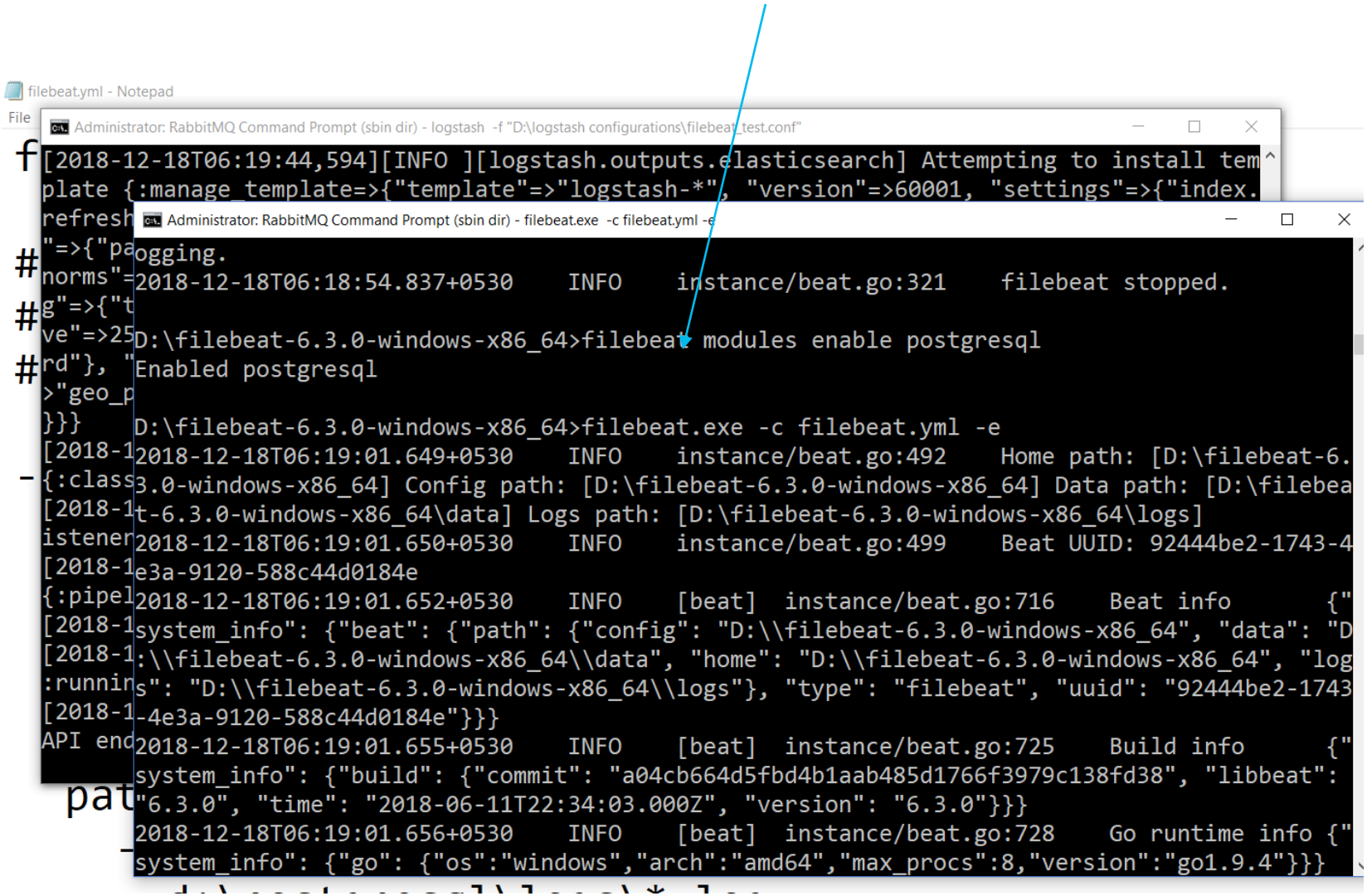
File Beat Configuration

```
Module system is already enabled

D:\filebeat-6.3.0-windows-x86_64>filebeat modules list
Enabled:
system

Disabled:
apache2
auditd
icinga
iis
kafka
logstash
mongodb
mysql
nginx
osquery
postgresql
redis
traefik

D:\filebeat-6.3.0-windows-x86_64>
```

File Beat Configuration

```
traefik

D:\filebeat-6.3.0-windows-x86_64>filebeat modules enable mysql
Enabled mysql

D:\filebeat-6.3.0-windows-x86_64>filebeat modules list
Enabled:
mysql
system

Disabled:
apache2
auditd
icinga
iis
kafka
logstash
mongodb
nginx
osquery
postgresql
redis
```

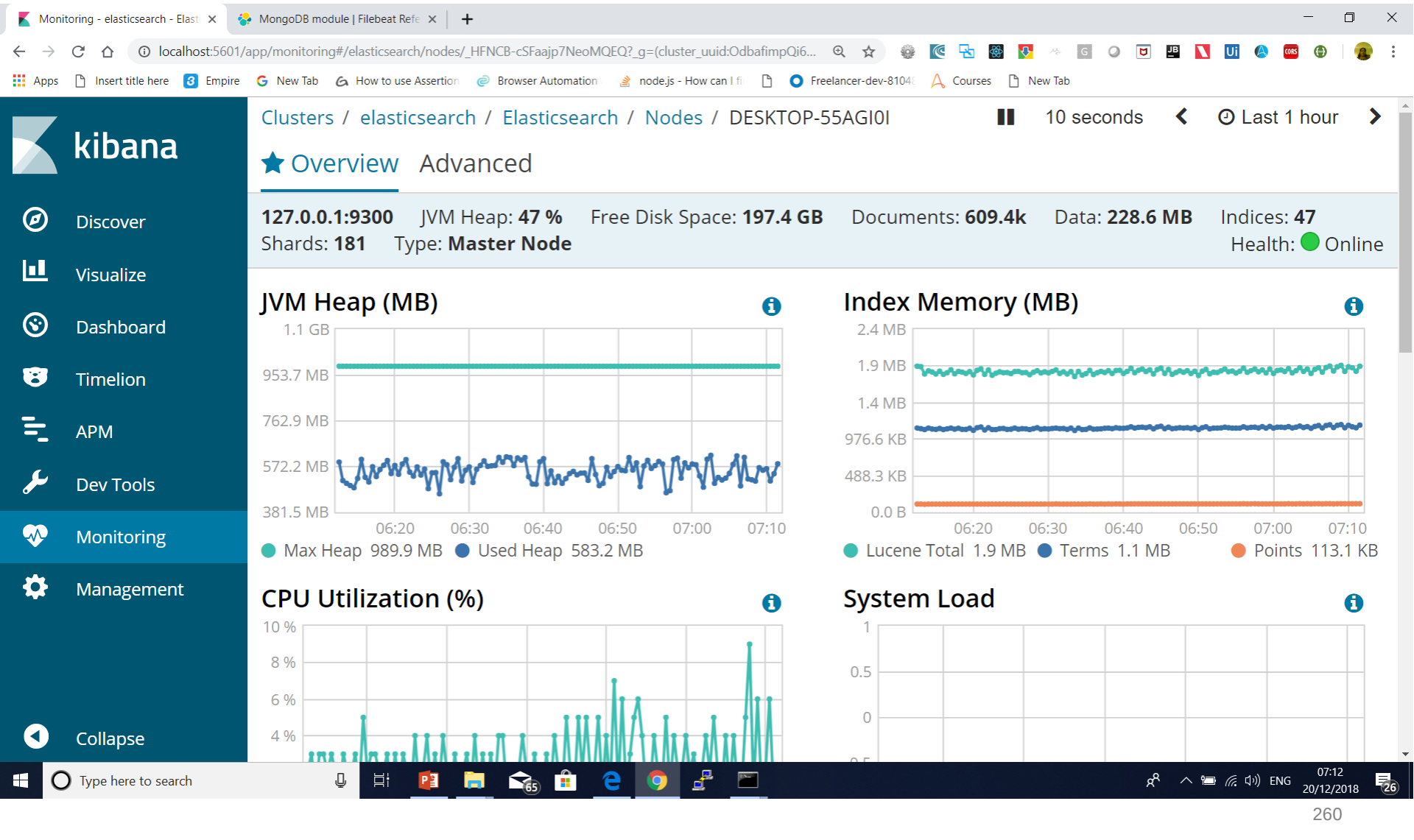
File Beat Configuration



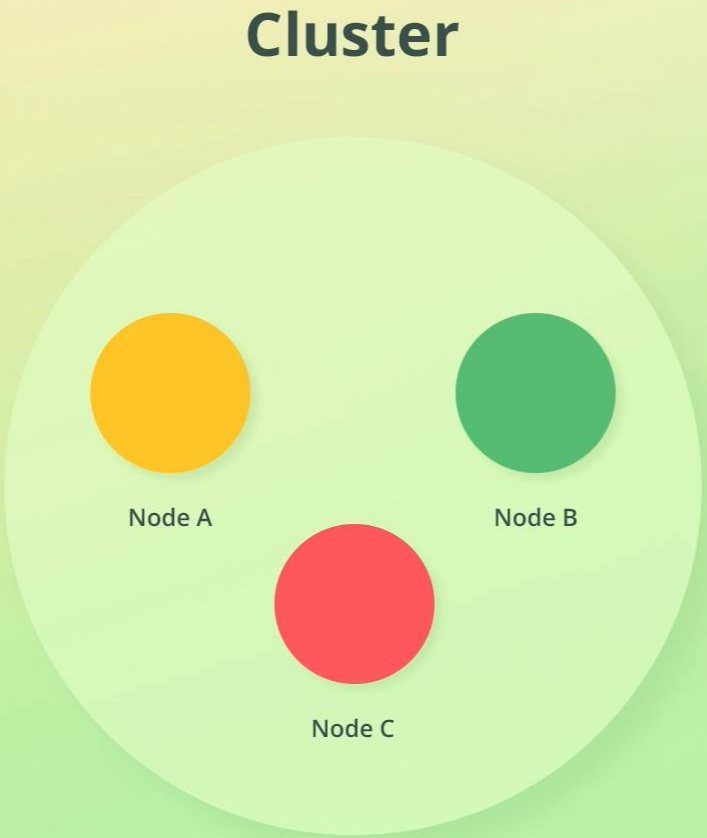
Running Filebeat

- filebeat.exe –modules mongodb -c filebeat.yml -e

Metric Beat


Monitoring - elasticsearch - Elast
MongoDB module | Filebeat Refe
localhost:5601/app/monitoring#/elasticsearch/nodes/_HFNCB-cSFaajp7NeoMQEQ?_g=(cluster_uuid:OdbafimpQi6...
Apps
Insert title here
Empire
New Tab
How to use Assertion
Browser Automation
node.js - How can I fi
Freelancer-dev-8104
Courses
New Tab
kibana
Discover
Visualize
Dashboard
Timelion
APM
Dev Tools
Monitoring
Management
Collapse
Clusters / elasticsearch / Elasticsearch / Nodes / DESKTOP-55AGI0I
10 seconds
Last 1 hour
Overview
Advanced
127.0.0.1:9300 JVM Heap: 47 % Free Disk Space: 197.4 GB Documents: 609.4k Data: 228.6 MB Indices: 47
Shards: 181 Type: Master Node
Health: Online
JVM Heap (MB)
1.1 GB
953.7 MB
762.9 MB
572.2 MB
381.5 MB
06:20
06:30
06:40
06:50
07:00
07:10
Max Heap 989.9 MB
Used Heap 583.2 MB
Index Memory (MB)
2.4 MB
1.9 MB
1.4 MB
976.6 KB
488.3 KB
0.0 B
06:20
06:30
06:40
06:50
07:00
07:10
Lucene Total 1.9 MB
Terms 1.1 MB
Points 113.1 KB
CPU Utilization (%)
10 %
8 %
6 %
4 %
System Load
1
0.5
0
Type here to search
ENG
07:12
20/12/2018

Elastic Cluster



Document

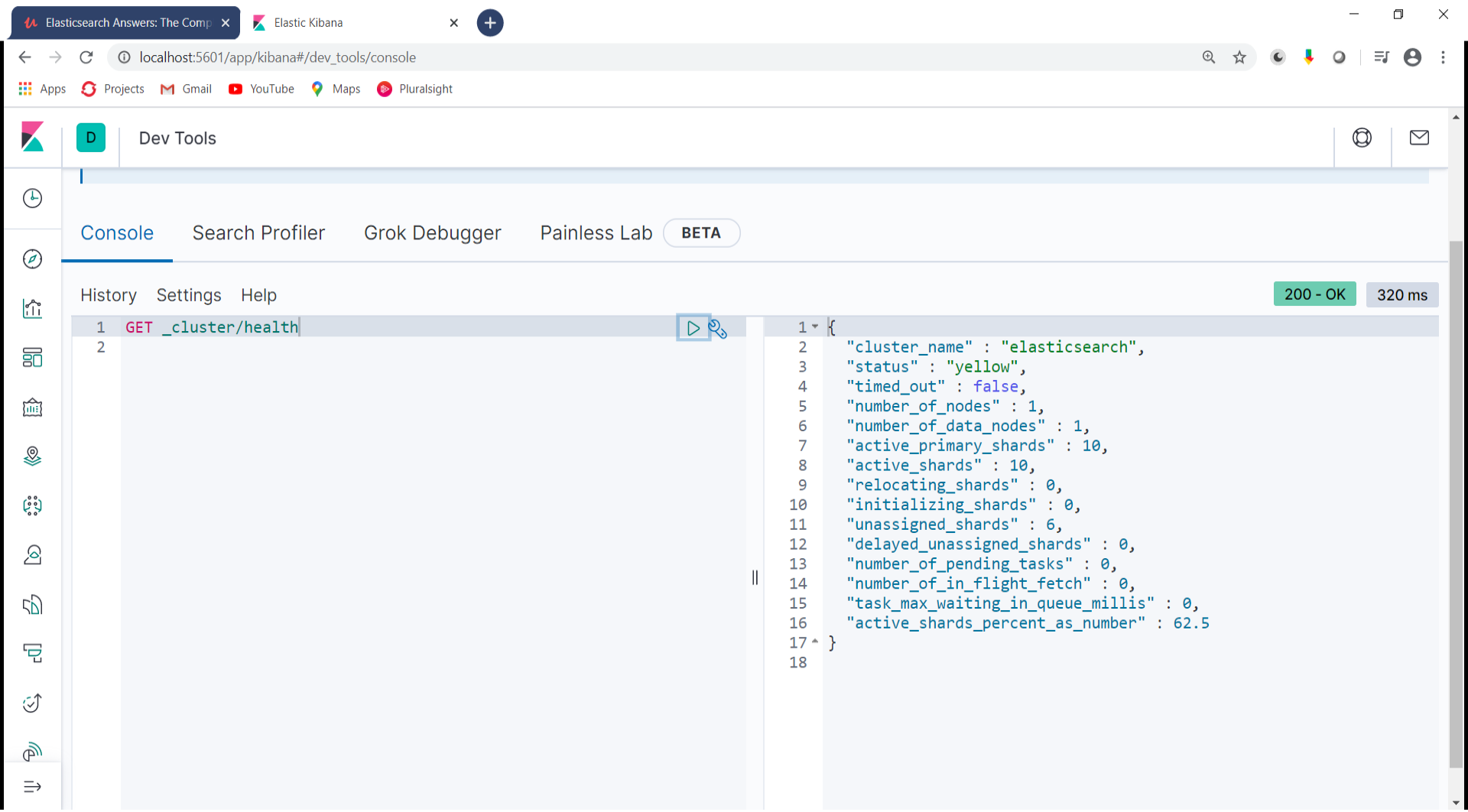
```
{
  "name": "Bo Andersen",
  "country": "Denmark"
}
```

is stored as

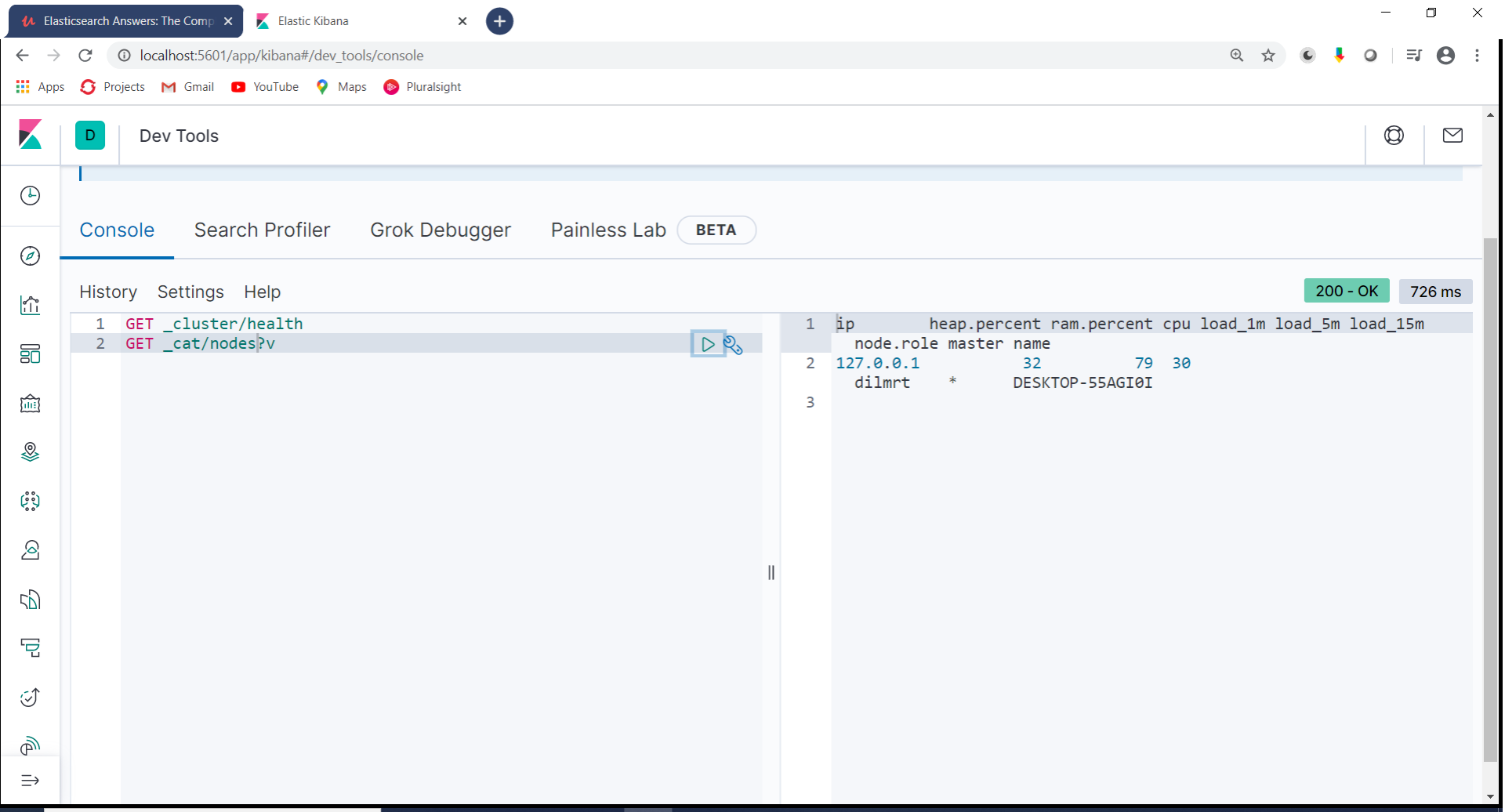
```
{
  "_index": "people",
  "_type": "_doc",
  "_id": "123",
  "_version": 1,
  "_seq_no": 0,
  "_primary_term": 1,
  "_source": {
    "name": "Bo Andersen",
    "country": "Denmark"
  }
}
```

Document and Indexes

GET _cluster/health



GET _cat/nodes?v



Elasticsearch Split Brain Problem

- Make sure nodes can communicate with each other quickly enough and that you won’t have a split brain.
- Split brain happens when two parts of the cluster that can’t communicate and think the other part dropped out.
- Suppose you have 2 nodes — Node 1 and Node 2 and you have just one index deployed.
- Node 1 stores the primary shard and Node 2 stores the replica shard. Node 1 gets elected as a master during cluster start-up.

Elasticsearch Split Brain Problem

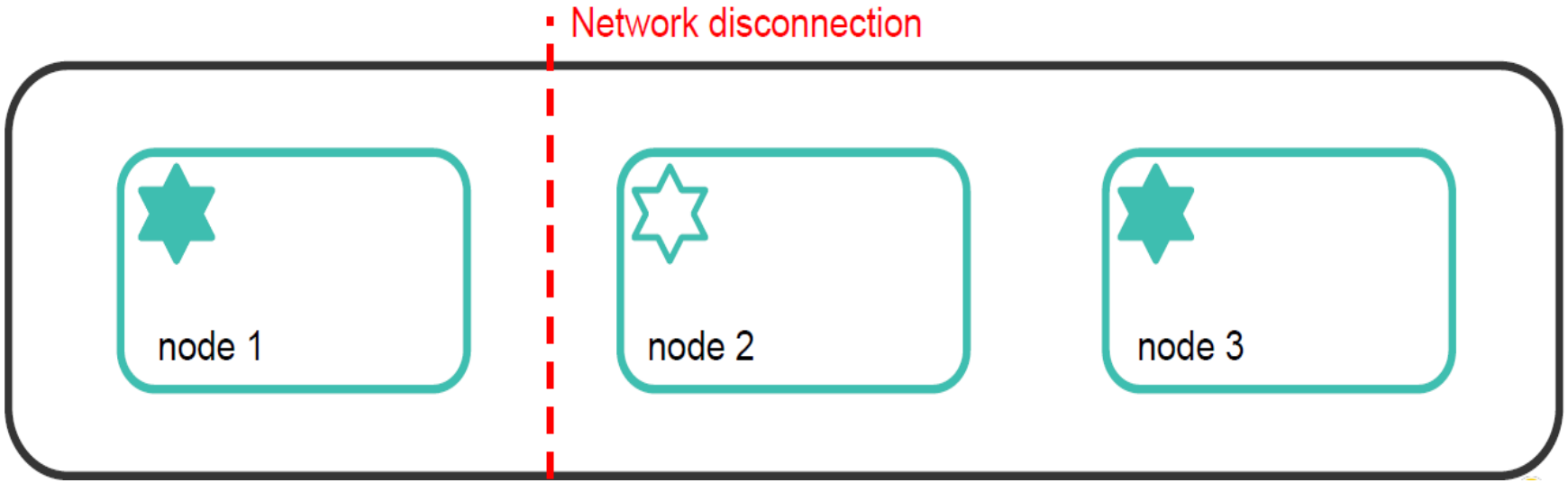
- Then suppose there is communication failure between the two nodes.
- Now, each of the nodes are in dark about the status of the other node and hence, they believe that the other has failed.
- Node 1 being the master will do nothing because it thinks it is up while the slave is down, so no issues.
- But Node 2 thinks that master has gone down and so is the primary shard, so it will automatically elect itself as master and promote the replica shard to a primary shard..

Elasticsearch Split Brain Problem

- Now, the cluster gets into a confused zone and can result in an inconsistent state.
- Indexing requests that will hit Node 1 will index data in its copy of the primary shard, while the requests that go to Node 2 will fill the second copy of the shard.
- This can result in situations like when searching for data: depending on the node the search request hits, results will differ.

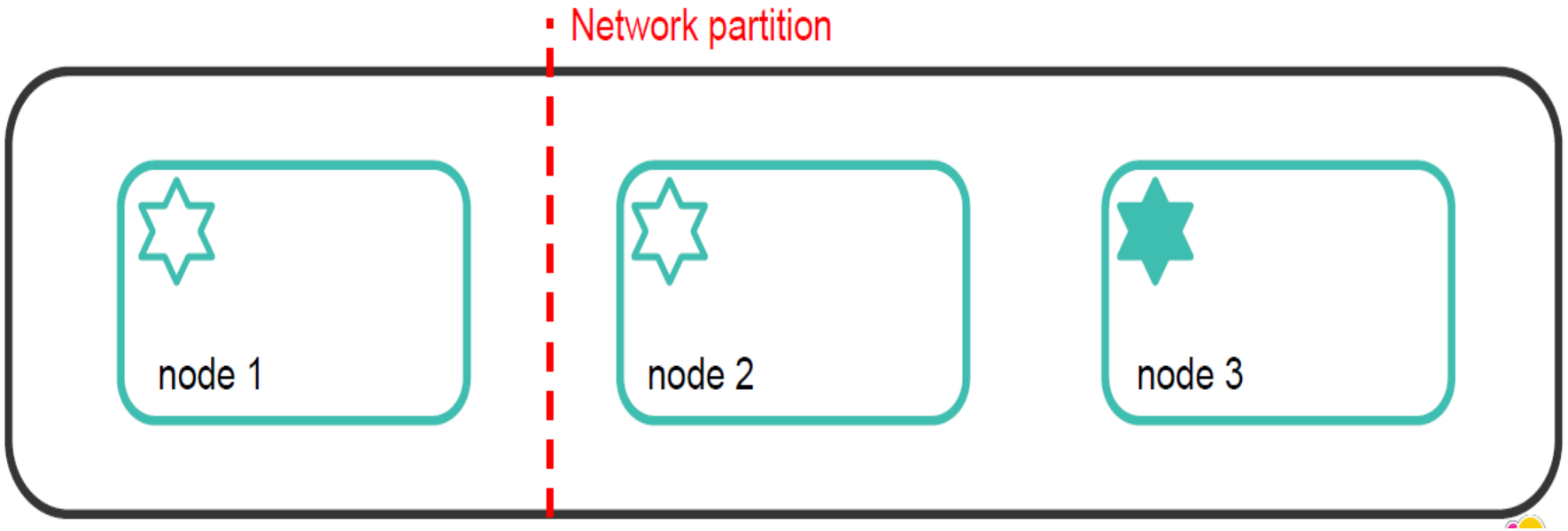
Split Brain

- Cluster with 3 master eligible nodes
- Concern if network becomes partitioned
- The cluster would inadvertently elect two masters, which is referred to “split brain”

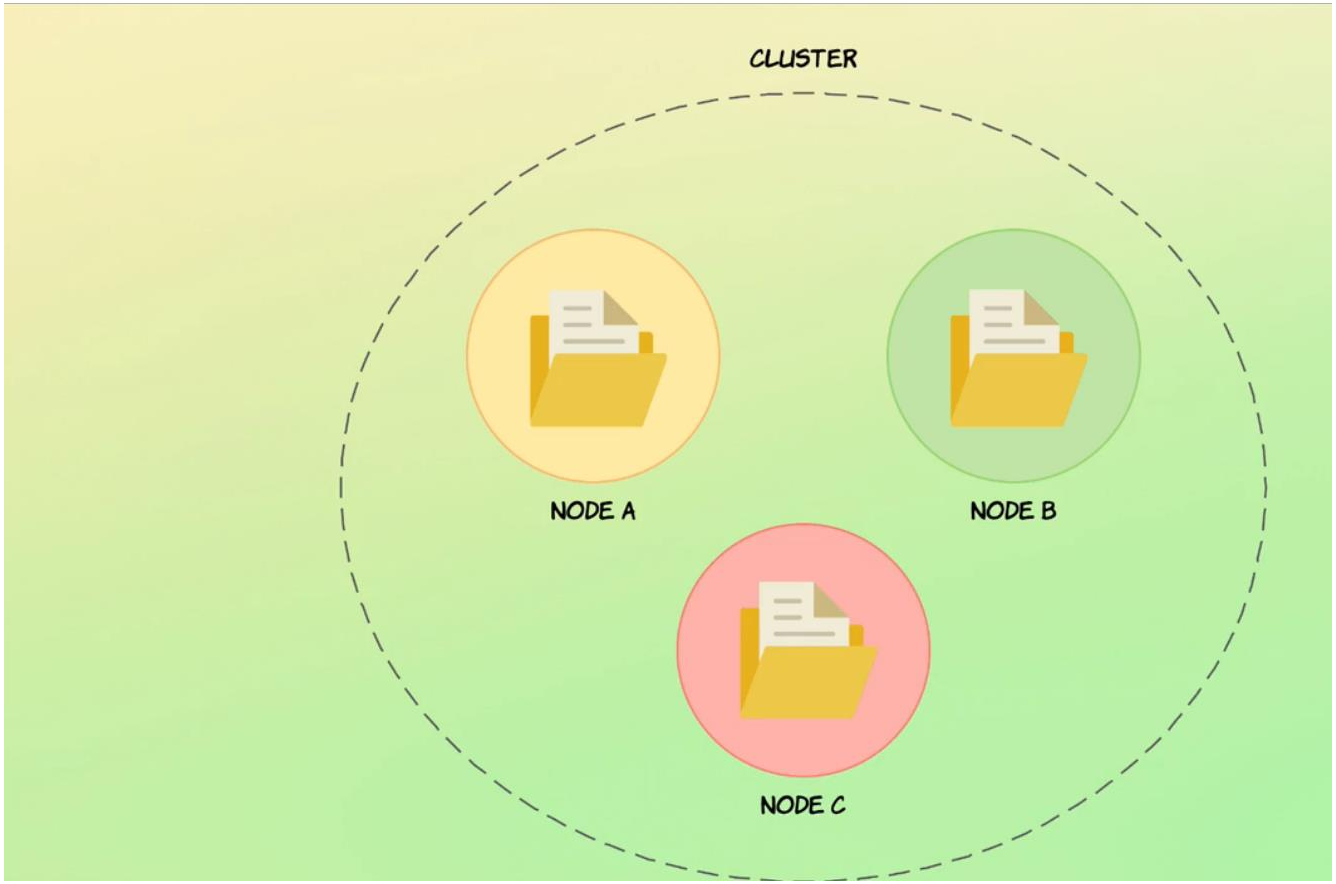


Avoiding Split Brain

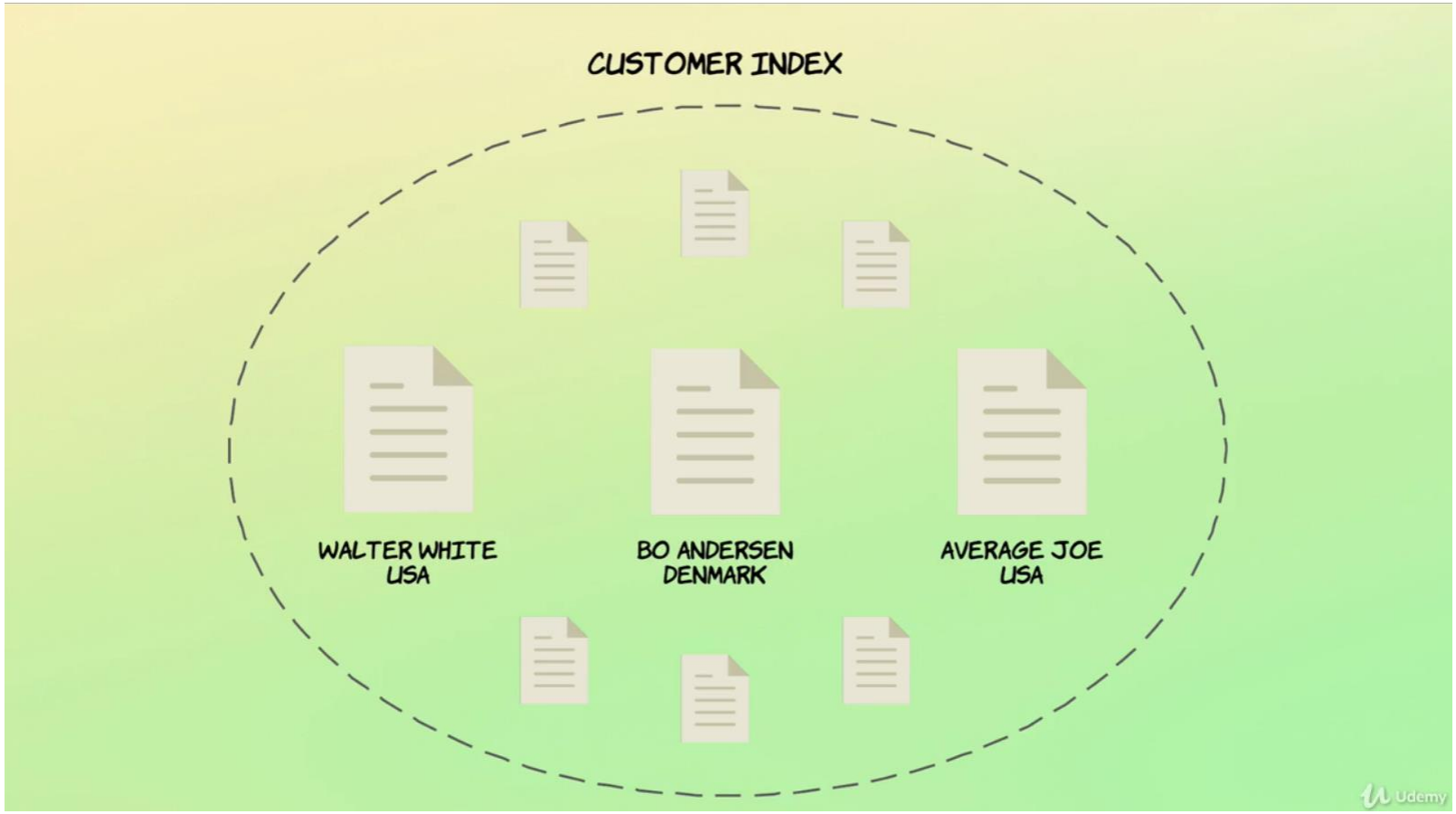
- A master eligible node needs at least minimum_master_nodes votes to win an election
  - Setting it to a quorum prevents the split brain scenario
- Recommendation for production clusters is to have 3 dedicated master eligible nodes
  - with the setting `minimum_master_nodes = 2`



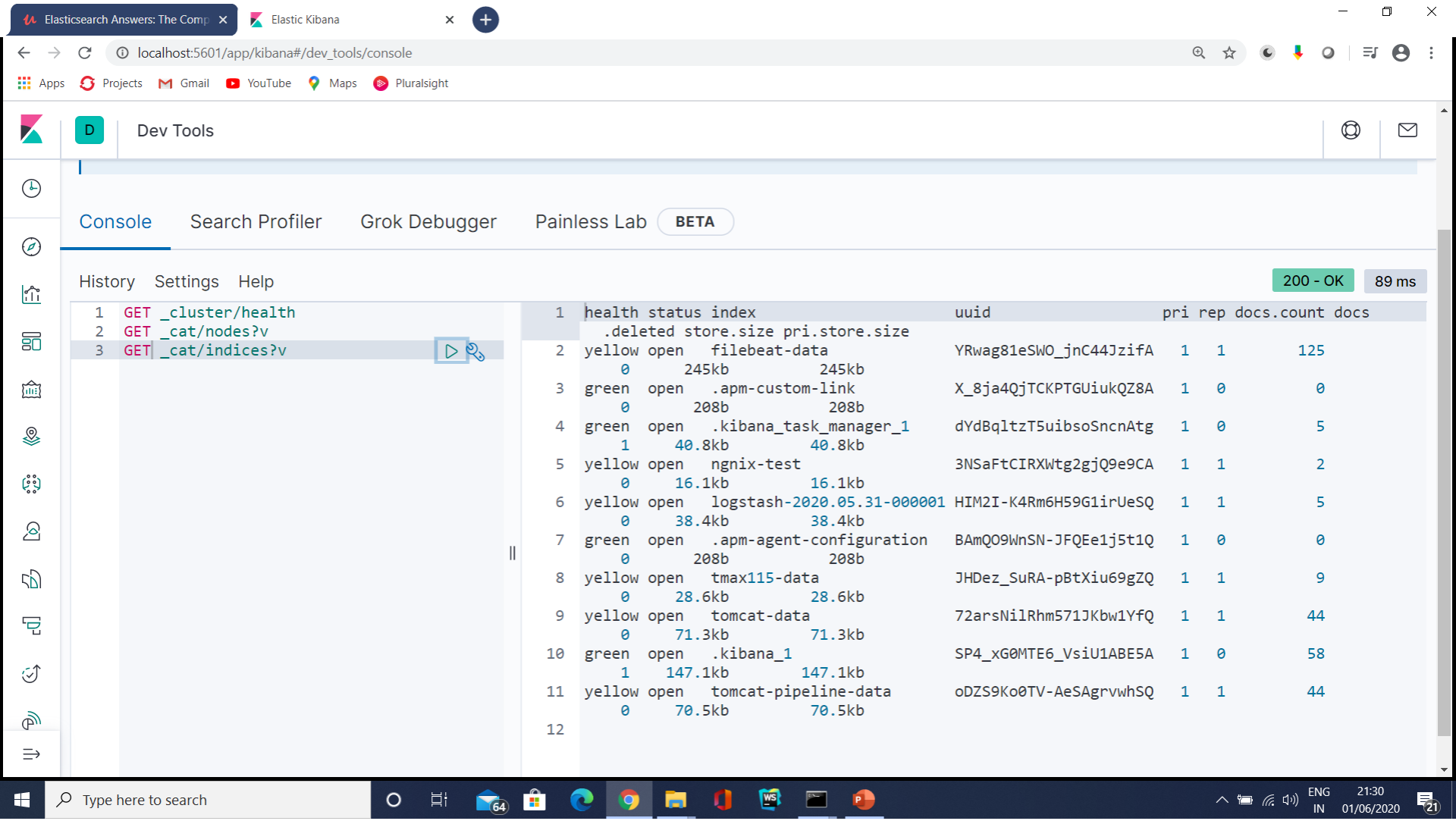
Nodes and Clusters



Nodes and Clusters



GET _cat/indices?v



Nodes and Clusters

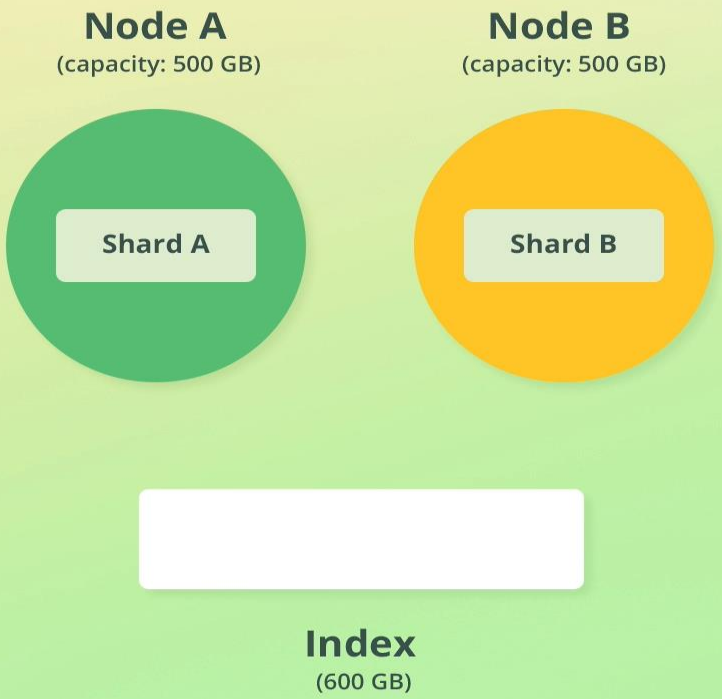
- What is an index?
- An index is stored in a set of shards, which are themselves Lucene indices.
- This already gives you a glimpse of the limits of using a new index all the time: Lucene indices have a small yet fixed overhead in terms of disk space, memory usage and file descriptors used.
- For that reason, a single large index is more efficient than several small indices: the fixed cost of the Lucene index is better amortized across many documents.

What is shard

- Sharding is a way to divide indices into smaller pieces
- Each piece is referred to as a *shard*
- Sharding is done at the index level
- The main purpose is to horizontally scale the data volume

What is shard

Index needs 600 GB space. Divide that to two different shards



What is shard

- A shard is an independent index... *kind of*
- Each shard is an Apache Lucene index
- An Elasticsearch index consists of one or more Lucene indices
- A shard has no predefined size; it grows as documents are added to it
- A shard may store up to about two billion documents

Purpose of Sharding

- Mainly to be able to store more documents
- To easier fit large indices onto nodes
- Improved performance
    - Parallelization of queries increases the throughput of an index

Configuring number of shards

- An index contains a single shard by default
- Indices in Elasticsearch < 7.0.0 were created with five shards
  - This often led to *over-sharding*
- Increase the number of shards with the Split API
- Reduce the number of shards with the Shrink API

How many shards are optimal

- There is no formula that will yield a number for us 😕
- There are many factors involved, so it *depends*
- Factors include the # of nodes and their capacity, the # of indices and their sizes, the # of queries, etc.
- Anticipate millions of documents? Consider adding a couple of shards
- Need to store some thousand documents? Stick to the default settings

How many shards are optimal

- There is no formula that will yield a number for us 😕
- There are many factors involved, so it *depends*
- Factors include the # of nodes and their capacity, the # of indices and their sizes, the # of queries, etc.
- Anticipate millions of documents? Consider adding a couple of shards
- Need to store some thousand documents? Stick to the default settings

Nodes and Clusters

- **How to search data.**
- While each shard is searched independently, Elasticsearch needs to merge results from all the searched shards.
- For instance, if we search across 10 indices that have 5 shards each, the node that coordinates the execution of a search request will need to merge 5x10=50 shard results.
- If there are too many shard results to merge and/or if you ran a heavy request that produces large shard responses (which can easily happen with aggregations), the task of merging all these shard results can **become very resource-intensive, both in terms of CPU and memory**.

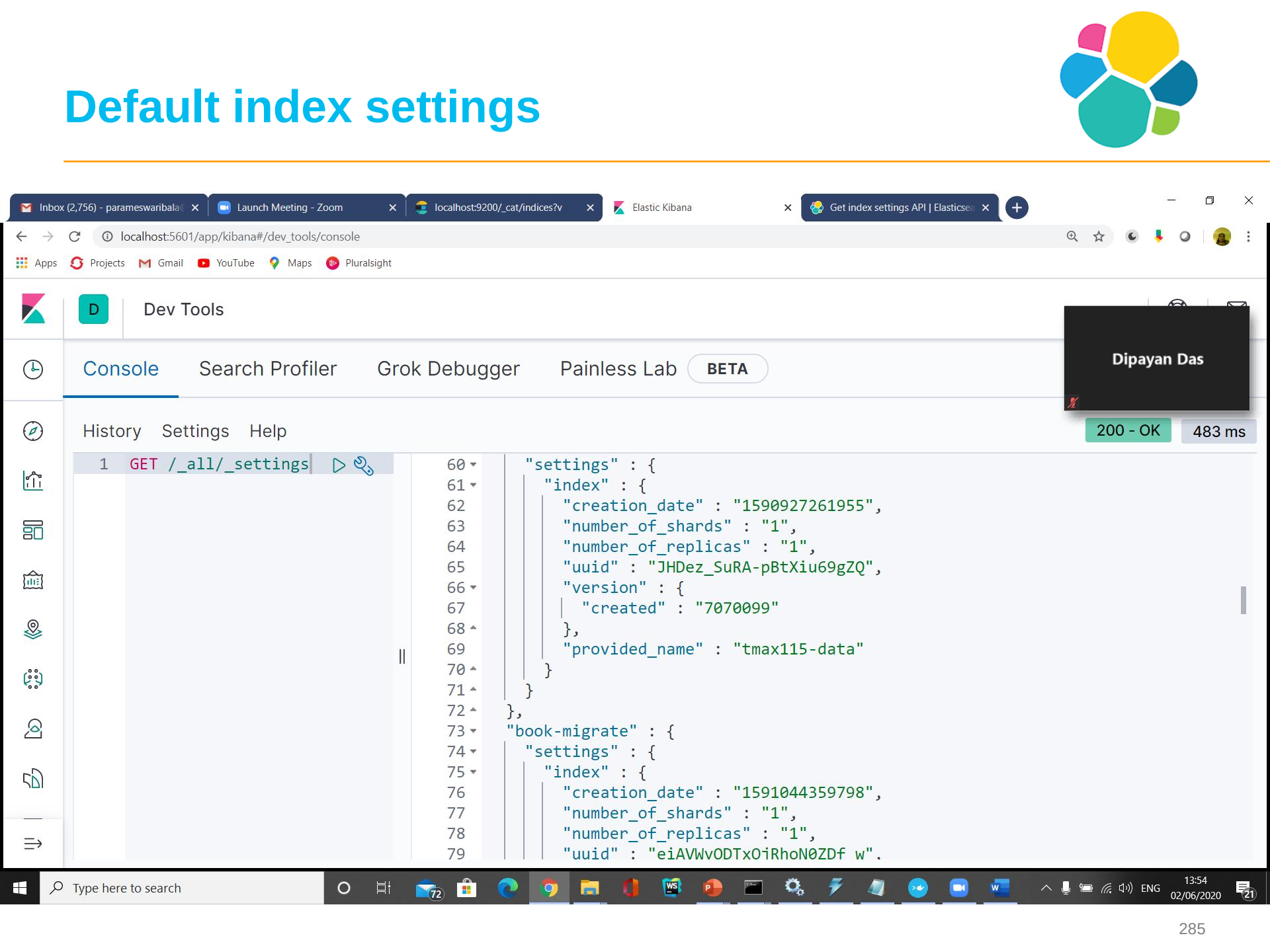
Shards

- The shard is the actual physical area where documents are stored.
- The index is just a logical namespace that references one or more shards.
- Applications need not be bothered with the details of shards, and they can perform all operations using the index.

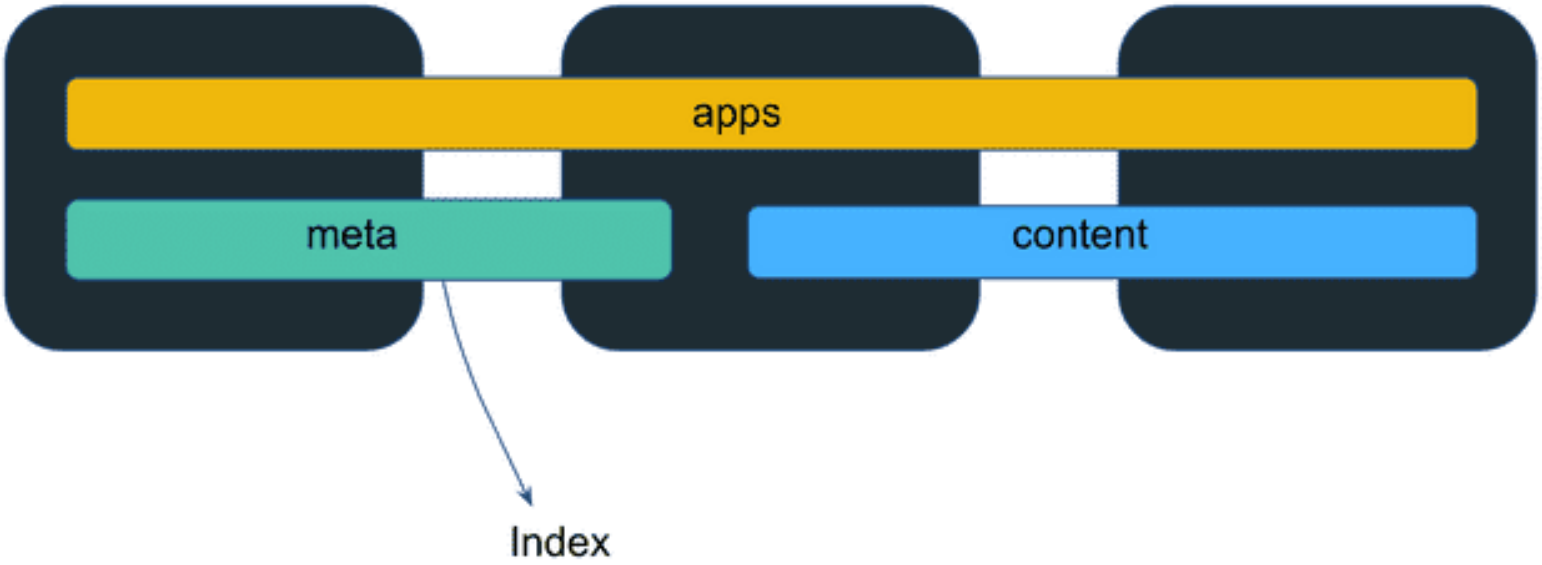
Shards(Primary and Replica)

- In order to maintain the availability of data in an Elasticsearch cluster, all documents are stored on one **primary** shard and multiple **replica** shards.
- When a document is indexed, it is first stored in its primary shard and then on corresponding replica shards.
- The default number of primary shards is **5** and it can be configured as per your needs.
- Replica shards generally reside on a node different from the primary shard.
- Replica shards load balance requests to take care of a high load and play a key role in case of failover.

Default index settings

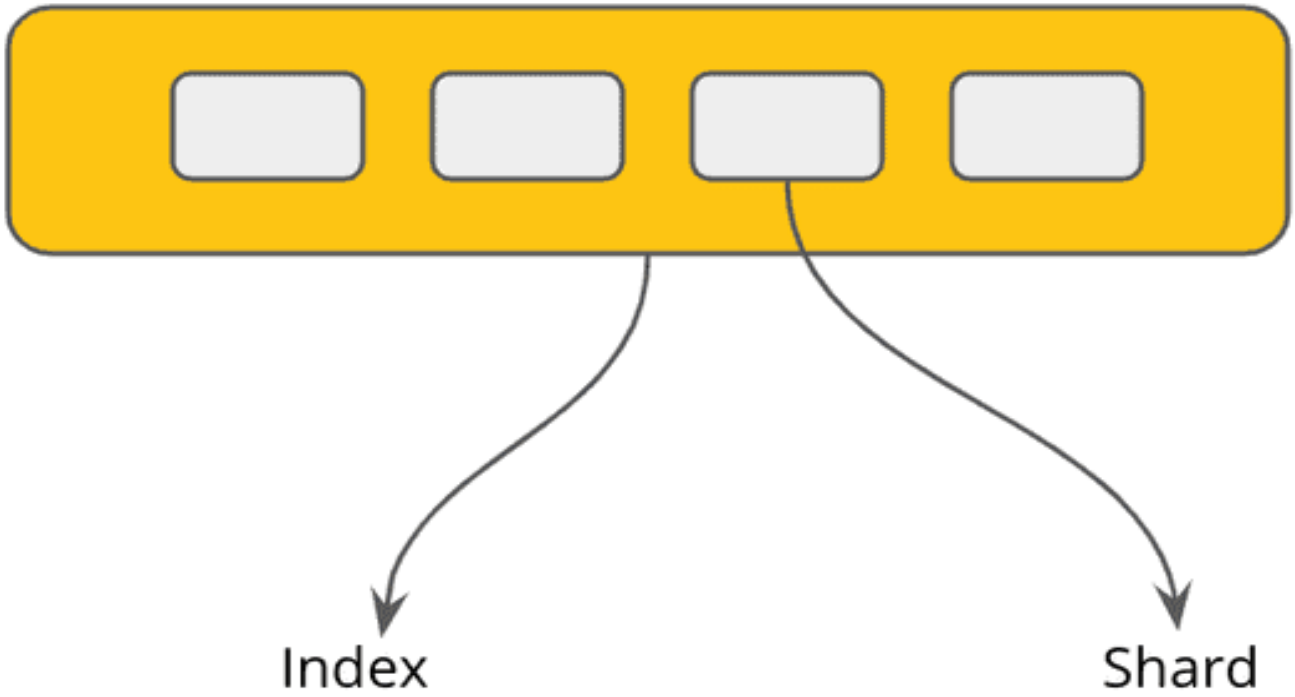


Index



Elasticsearch is a distributed Search engine. The data is stored and spread across various nodes. Indexes are the ones that hold that data logically (physical data is in shards).

Index



Index

RDBMS with SQL

```
select * from emp-index
```

Elasticsearch with QueryDSL

Index name

```
GET emp-index/_search
{
    "query": {
        "match_all": {}
    }
}
```

Index Naming Rules

- Lowercase only
- Cannot include \, /, ``, ?, ", <, >, |, (space character), ,, #
- Indices prior to 7.0 could contain a colon (:), but that has been deprecated and will not be supported in 7.0+
- Cannot start with ``, _, +
- Cannot be . or ..
- Cannot be longer than 255 bytes (note it is bytes, so multi-byte characters will count towards the 255 limit faster)

Index Naming Strategy

- For time-series data such as logs, metrics, traces., Elastic's Beats or Logstash, write data into Elasticsearch with default index names like filebeat-0001 or logstash-%{+yyyy.MM.dd}.
- It automatically increments the index name after a specific data limit.
- These defaults also help in managing data with other features like Index/Snapshot Lifecycle Management Policies.
- If one has a strong opinion about creating indices with naming strategies, go ahead to create your own, but think about the scaling strategy.
- Keeping with defaults helps one can correlating the data (ECS) easily and leverage all the inbuilt features.

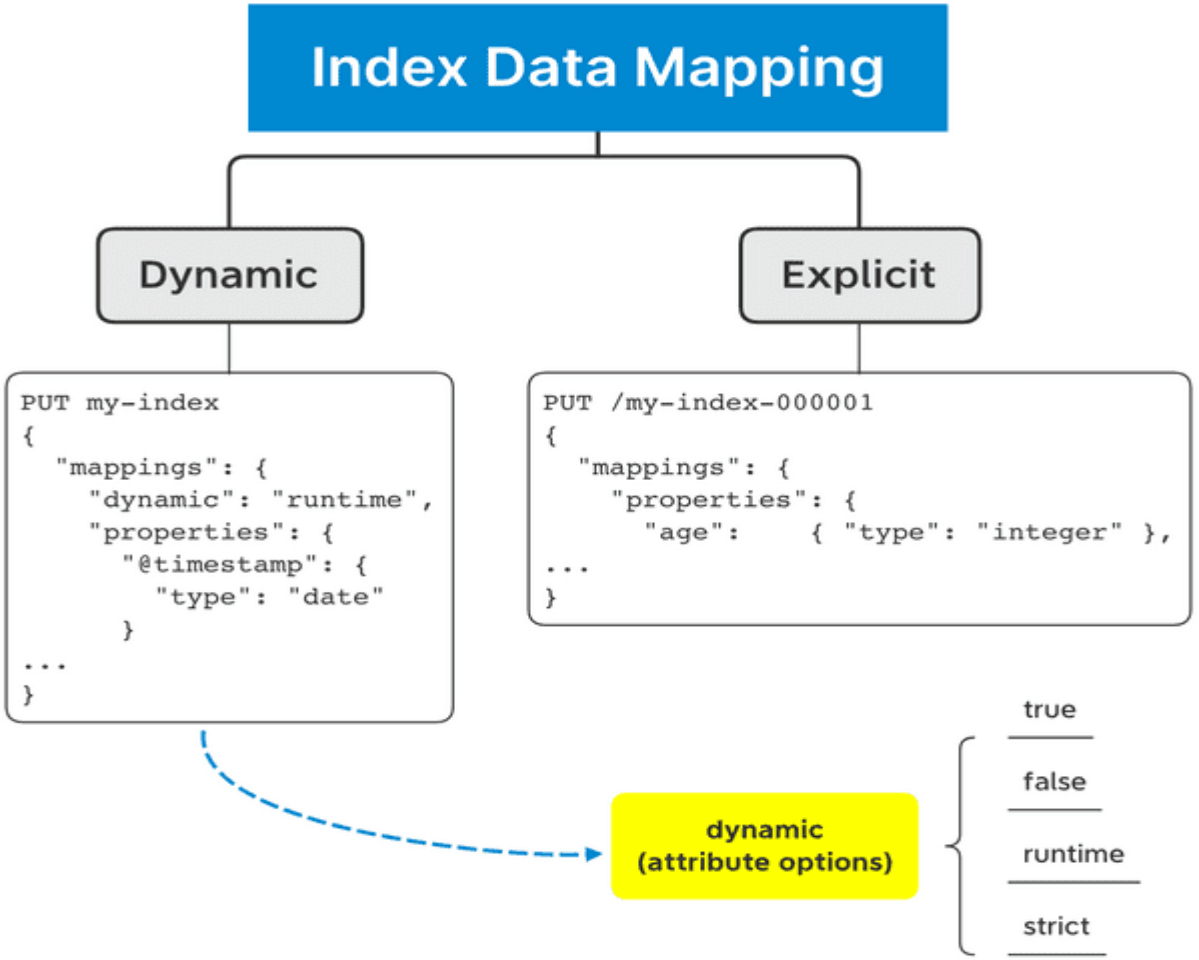
Categories of Data Stored in indexes

- Indices store different data categories, although, remember, each data has a different volume, tenure.
  - Time series (Data indexed in the order of time)
  - Logs
  - Metrics
  - Traces
  - Security Events
  - Content
  - PDFs, Spreadsheets, PPT's etc
  - Data from RDBMS

Index Data Mapping

- Elasticsearch prepares the data for search/query operations while ingesting the data.
- It is also called "Schema on Write."
- It is like creating the schema dynamically/explicitly while defining inserting the record in a SQL Table.
- One can also do "Schema on Read," which means preparing or fetching the data while querying with Runtime fields.

Index Data Mapping



Dynamic

- It happens on the fly as and when we ingest the data. It can be set to false, true, runtime, strict.
- It means one can enable or disable the dynamic mapping even make it strict to follow a specific set of schema, otherwise reject the client's documents for ingestion.

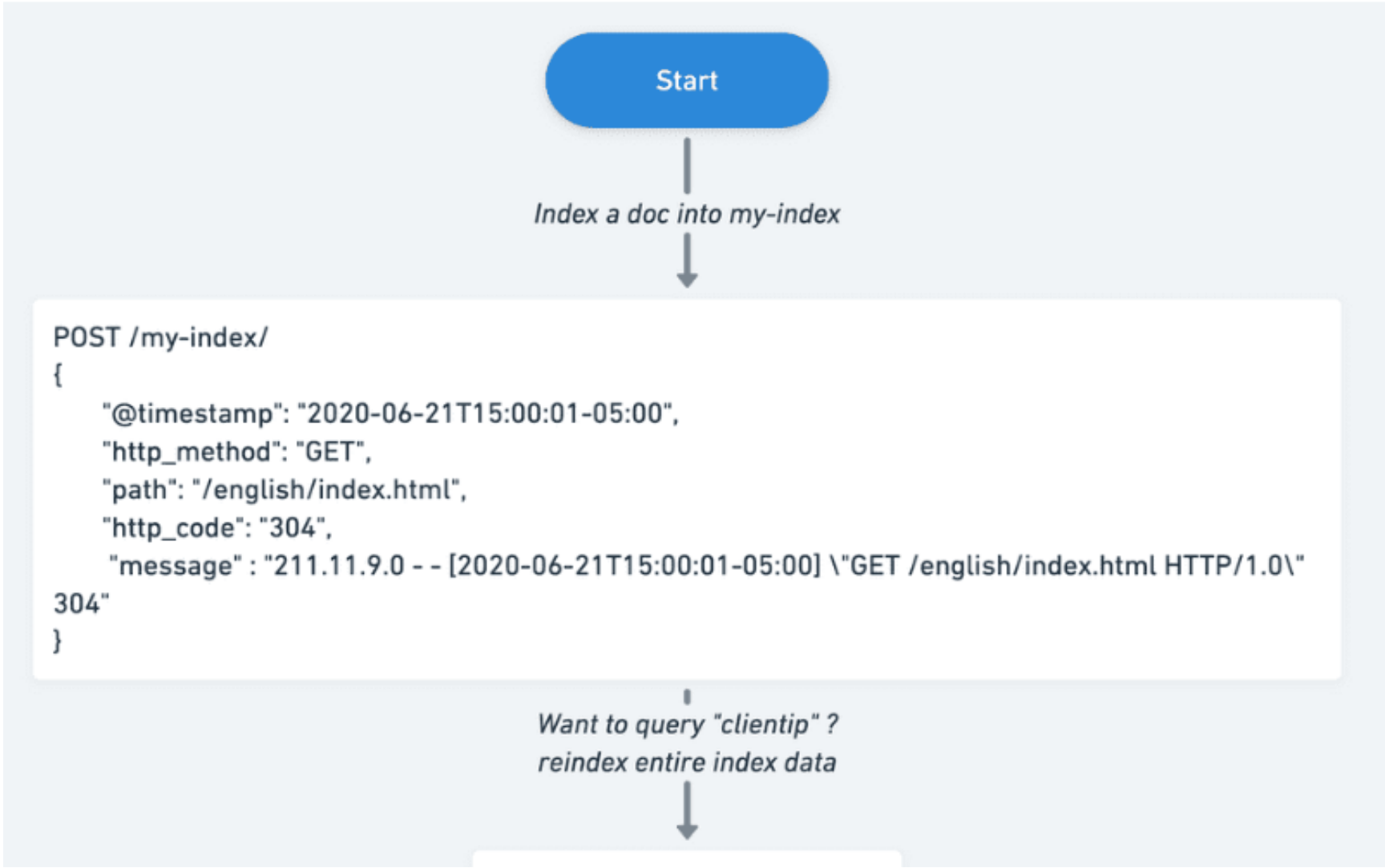
Explicit

- Can be set like an RDBMS Table while creating the index.
- Once defined, it can only be changed based on the mapping parameters.
- If needed to change, data needs to be reindexed.
- Method 1: One could use an alias for a field name and swap to new fields, deprecating the documents' old field.
- Method 2: Create another index with the desired field data type + existing fields in the document, field names, then reindex the data.
- Elasticsearch attempts to index all the fields client sends, but sometimes if we don't want specific fields to be indexed, set enabled field to false and those fields will not be prepared for querying (can also be called not-indexed, unindexed).

Runtime Fields:

- Elasticsearch uses the "Schema on Write" methodology, which enables faster querying, better visualizations.
- However, if unindexed data in the _source needs to be queried later, you need to reindex to bring that data into a field.

Runtime Fields:



Runtime Fields:

Runtime Fields:

```
GET my-index/_search
{
  "size": 1,
  "query": {
    "match": {
      "clientip": "211.11.9.0"
    }
  },
  "fields" : ["*"]
}
```

Templates

- There are two types of Templates in Elasticsearch: Index Templates, Component Templates.

- Index templates are settings that get applied when an index with a specific pattern gets created, for example, logs-* or app-*.