Elasticsearch and Kibana

Stuti Chaturvedi
202161006

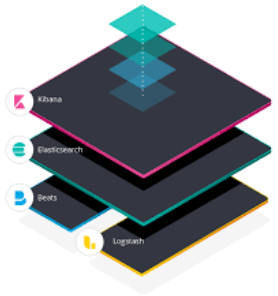
Haripriya Goswami
202161003

Yashita Vajpayee
202162012

(cm)0.4pt
***Abstract*—In this experiment, we have experimented sample codes on Elasticsearch and Kibana**

I. Introduction : Elasticsearch and Kibana

Elasticsearch and Kibana are the components of the ELK stack (owned by Elastic) where ELK stands for Elasticsearch, Logstash, and Kibana. It allows users to input data in any format and process, analyze, and visualize it in real-time. Additionally, Centralized logging can be done with the help of this stack to determine problems with applications or servers.
Real-time examples of ELK stack are:
Netflix: It heavily depends on the ELK stack for monitoring and analyzing customer service operations' security logs by allowing them to index store, and search documents from about 15 clusters comprising of nearly 800 nodes.
LinkedIn: It uses the ELK stack for monitoring performance and security. The IT team has integrated ELK with Kafka to support its real-time overhead.
We will be looking at Elasticsearch and Kibana in the following report.



A. Elasticsearch

A distributed, open-source, real-time, and analytics engine developed in Java. It is based on a search engine named Lucene and built with RESTful APIs. It utilizes structures based on documents rather than tables and schemas. It allows us to maintain and analyze a gigantic volume of information in real-time. Elasticsearch works with JSON types of document files. It facilitates us by parsing our data to retrieve desirable information in real-time. It is primarily used as the underlying mechanism to power applications that concluded search requirements. It has been embraced in search engine platforms for modern web and mobile applications.

Elasticsearch offers top reliability, straightforward deployment, and effortless management. It also offers progressive queries to perform thorough analysis and stores all the data centrally. It is helpful to query the documents quickly. The tool also offers complex analysis and several advanced features added to the quick search.
To work with Elasticsearch, there are a few terms to consider:
Query: A query can be depicted as a request to a database for information or data. The language to execute and integrate many types of searches like structured, unstructured, etc.
Cluster: A cluster is a group of nodes that together control data and provide joined indexing and search capabilities.
Index: An index is an assemblage of documents having similar features. It is very beneficial while performing indexing, search, update, and delete operations. It permits us to demarcate as many indexes in a cluster.
Document: It is the fundamental unit of information that is capable of indexing. It can be represented in JSON form of (key: value) pair. ‘“item”: “null”’. Each document is related with a unique id and a type.
Shard: Each index can be broken into a number of shards in order to distribute data. The shard is an atomic component of any index that can be dispersed over the cluster when more nodes have to be added.
Why Elasticsearch?

- Stores schema-less data and are able to create a schema for your data
- Filter and query data for insights
- Offers horizontal scaling, reliability, and multitenant capability for indexing in real-time to make it search faster.
- Able to scale both vertically as well as horizontally

B. Kibana

Kibana is a data visualization tool that is helpful in visualizing the queries executed in Elasticsearch documents and thus has brief insights into the processed data. The Kibana Dashboard facilitates the user with various geospatial data and interactive diagrams and graphs to visualize the queries. It is useful for searching, viewing, and interacting with the data stored in the Elasticsearch documents. It also helps in performing advanced analysis on data and visualizing it in various forms such as tables, charts, and maps.

Kibana has different methods to search data. The most common ones are-
Free text searches: useful to search a specific string.
Field level searches: useful to search a string within a specific field
Logical statements: useful in combining searches into logical statements.
Proximity searches: useful to search terms within specific character proximity.

Why Kibana?

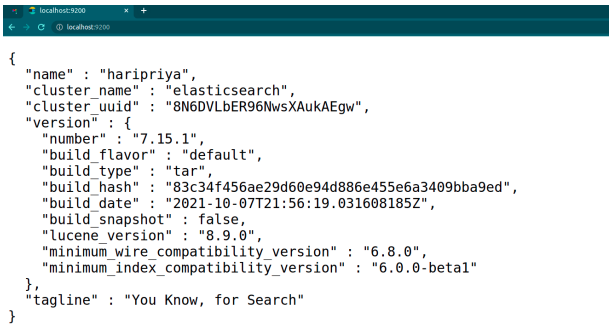
- Easy visualization
- Totally integrated with Elasticsearch
- Offers real-time analysis, charting, outlining, and debugging capabilities.

II. Implementation

First, we have to run the local server in order to start working with Elasticsearch as well as Kibana

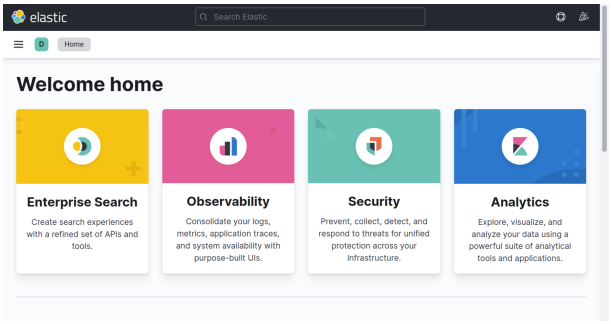
We can start the Elasticsearch by running the following code in terminal **bin/Elasticsearch**

Without Kibana, we can work with Elasticsearch by running queries locally and viewing outputs on the **localhost/9200**

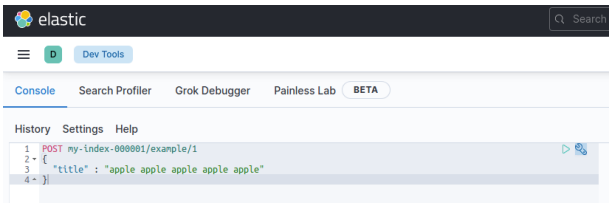


The same process can be run on the same interface in Kibana. To start the Kibana server, we can run **bin/Kibana** in the terminal similar to the Elasticsearch

Now, we can work on Kibana Dashboard that we talked about earlier as the following. We can open it on **localhost/5601**

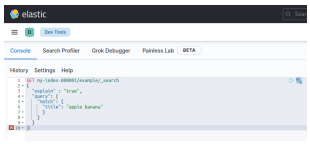


Following are the outputs of the sample exercise implemented in the Kibana Dev Tools



This is the query that we ran to add Documents (Example1, Example2, Example3, ..).

This is the output that we get for the same.
Now, we can query terms in Documents as follows:



By running the above query, we get the following output:

III. References

Introduction to information retrieval by Christopher D Manning Prabhakar Raghavan Hinrich Schutze

Elastic, 2021. Free and open search: The creators of Elasticsearch, Elk Kibana. Elastic. Available at: https://www.elastic.co/