# ELK Stack:

In the microservices world the challenge is that microservices are isolated among themselves and they do not share common database and log files. As the number of microservices increase and we enable cloud deployment with automated continuous integration tools, it becomes very complicated to check the problems and debugging. So, it is very much necessary to have some provision of debugging the components when we have any problem.

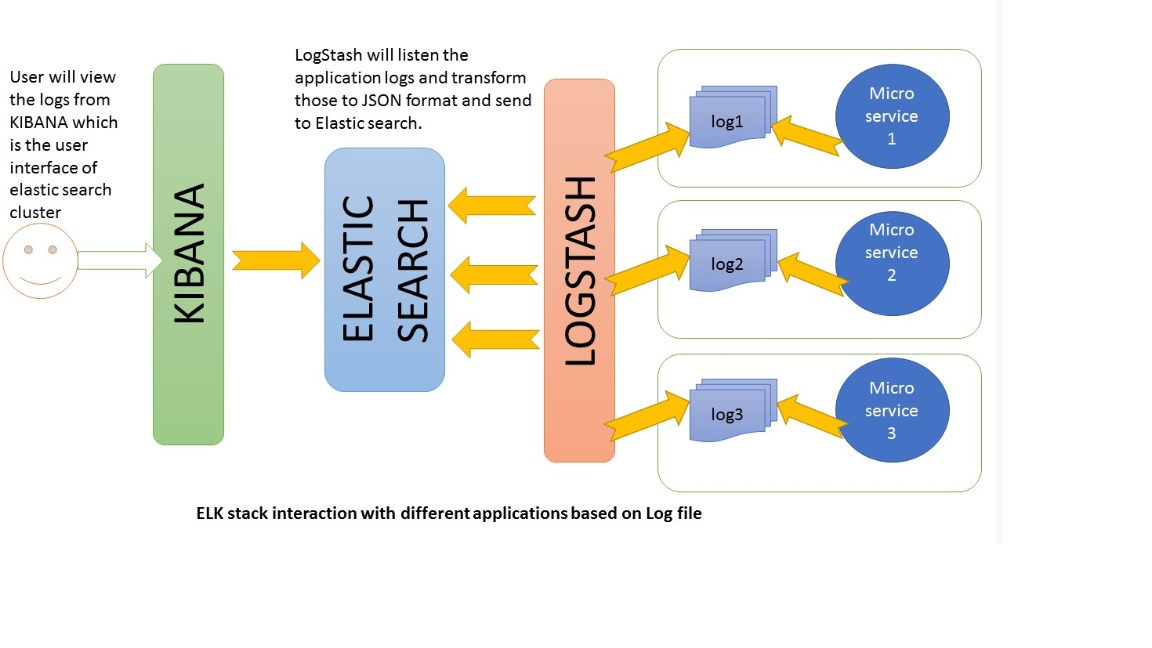
We have a set of tools which can solve the above problems when properly used together. One such popular set of tools are [**Elastic Search**](https://www.elastic.co/)**,** [**Logstash**](https://www.elastic.co/products/logstash) **and** [**Kibana**](https://en.wikipedia.org/wiki/Kibana) **– together referred as ELK stack**. They are used for searching, analyzing, and visualizing log data in a real time.

## What is ELK Stack

1. **Elasticsearch** is a NoSQL database which is based on Lucene search engine and is built with RESTful APIs. It is a highly flexible and distributed search and analytics engine. Also, it provides simple deployment, maximum reliability, and easy management through horizontal scalability. It provides advanced queries to perform detailed analysis and stores all the data centrally for quick search of the documents.
2. **Logstash** is the data collection pipeline tool. It the first component of ELK Stack which collects data inputs and feeds it to the Elasticsearch. It collects various types of data from different sources, all at once and makes it available immediately for further use.
3. **Kibana** is a data visualization tool. It is used for visualizing the Elasticsearch documents and helps the developers to have an immediate insight into it. Kibana dashboard provides various interactive diagrams, geospatial data, timelines, and graphs to visualize the complex queries done using Elasticsearch. Using Kibana you can create and save custom graphs according to your specific needs.

#### ELK Stack Architecture:

Logstash processes the application log files based on the filter criteria we set and sends those logs to Elasticsearch. Through Kibana, we view and analyze those logs when required.



#### **Note**: The below setup is Windows based:

#### Elasticsearch

* Download the latest version of Elasticsearch from the below link and unzip it any folder.

<https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-7.4.0-windows-x86_64.zip>

* Run bin\elasticsearch.bat from command prompt.
* By default, it would start at [http://localhost:9200](http://localhost:9200/)

#### Kibana

* Download the latest distribution from below link and unzip into any folder.

<https://www.elastic.co/downloads/kibana>

* Open config/kibana.yml in an editor and set elasticsearch.url to point at your Elasticsearch instance. In our case as we will use the local instance just uncomment elasticsearch.url: "<http://localhost:9200>"
* Run bin\kibana.bat from command prompt.
* Once started successfully, Kibana will start on default port 5601 and Kibana UI will be available at [http://localhost:5601](http://localhost:5601/)

#### Logstash

* Download the latest distribution from [download](https://www.elastic.co/downloads/logstash) from the below link and unzip into any folder.

<https://www.elastic.co/downloads/logstash>

* Create one file logstash.conf as per [configuration instructions](https://www.elastic.co/guide/en/logstash/current/configuration.html). We will again come to this point during the actual demo time for exact configuration.
* Now run the command

bin/logstash -f logstash.conf to start logstash

ELK stack is now up and running. And the Kibana UI can be accessed at <http://localhost:5601>.

Now we need to create few microservices and point Logstash to the API log path.

# Configuring Logstash

To configure Logstash, you create a config file that specifies which plugins you want to use and settings for each plugin. You can reference event fields in a configuration and use conditionals to process events when they meet certain criteria. When you run logstash, you use the -f to specify your config file.

Let’s step through creating a simple config file and using it to run Logstash. Create a file named "logstash-simple.conf" and save it in the same directory as Logstash.

input { stdin { } }

output {

elasticsearch { hosts => ["localhost:9200"] }

stdout { codec => rubydebug }

}

Then, run logstash and specify the configuration file with the -f flag.

bin/logstash -f logstash-simple.conf

Logstash reads the specified configuration file and outputs to both Elasticsearch and stdout. Note that if you see a message in stdout that reads "Elasticsearch Unreachable" that you will need to make sure Elasticsearch is installed and up and reachable on port 9200.

**Screen Shots of Local environment:**

