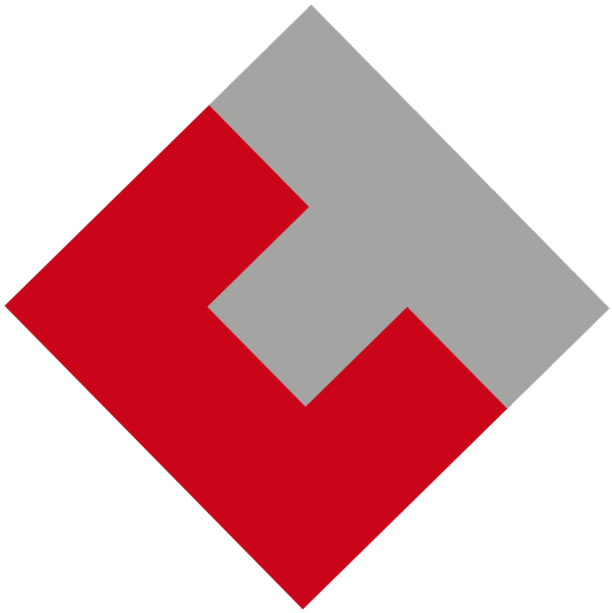


Coriolis Technologies

Summer Internship Report

Author:

Mohammed Hashim M

Chennai Mathematical Institute

MDS202124

Supervisor:

Sudhir Kumar

Tech Lead,

Coriolis Technologies

June - July 2022

Contents

1 Acknowledgement

My deep gratitude goes first to the Coriolis Management and the CEO Mr. Basant Rajan for providing me the opportunity to pursue an internship here. I am indebted to my mentors Mr. Sudhir Kumar and Mr. Rohan Nandode, who expertly guided me throughout my internship.

2 Abstract

The task is to create a kubernetes cluster to deploy Deep Learning models which is scalable and fault tolerant and automate the entire process end to end.

3 Introduction

We are streaming video frames 40 cameras and ML models are analyzing them.Currently 4 million data is crunched using deep learning models and the idea is to scale the ecosystem to crunch 15 millions of data per day.

Kubernetes is known to be scalable and fault tolerant clustering solution.Ansible is a suite of software tools that enables infrastructure as code. Make use of the ecosystem to automate the deployment of the Deep Learning models to make system performant and scalable.

4 Ansible Automation

Automation was implemented using ansible-playbook.Ansible Playbooks offer a repeatable, re-usable, simple configuration management and multi-machine deployment system, one that is well suited to deploying complex applications.

Main processes to automate were the following -

- Kubernetes dependencies installation
- Cluster initialisation by master node
- Cluster join by worker nodes
- Deployment of various components of cluster

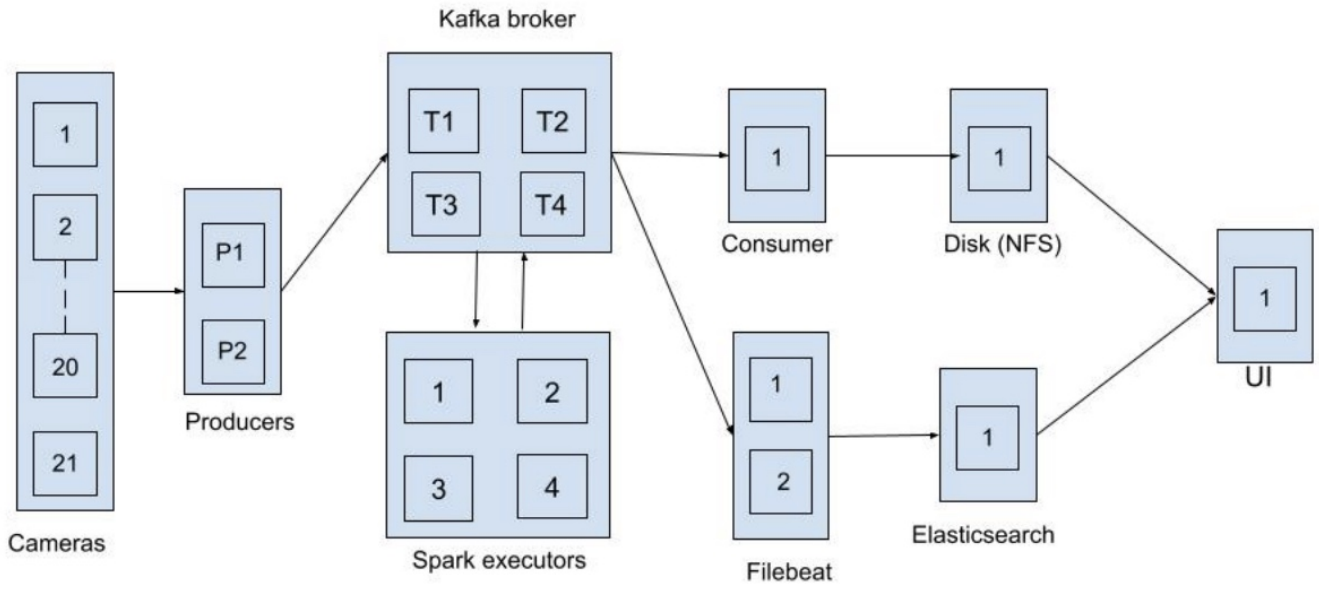
With the help of ansible-playbook script we could avoid manual process carried in each nodes.

5 Cluster

As shown in the Fig. 1, there are various components in the cluster. We describe them below:

- **Producers:** Producers take data coming from the cameras, perform some basic operations on the image and sends the data to Kafka broker.
- **Kafka and Zookeeper:** Kafka is a distributed data store that is used for storing and retrieving streaming data. Kafka doesn't natively support Kubernetes. So, we need to deploy Zookeeper in the cluster.
- **Spark drivers and executors:** The various ML models that were described earlier are deployed using spark. The various spark executors take data from kafka and send their output to the kafka broker.
- **Consumer:** The consumer takes the output from object detection model and saves it to the disk.
- **Filebeat:** This tool sends the output the spark executors (Deep Learning Models), which is is present in kafka broker to elasticsearch.
- **Elasticsearch:** Elasticsearch is a distributed search and analytics engine built on Apache Lucene. All the data that comes out of the ML models is saved in the elasticsearch database.
- **UI:** This is a user interface which displays images and result of Deep Learning models

Figure 1: Data flow diagram.



6 ConfigMaps

ConfigMap are the API object used to store data key-value pairs.The deployed pods consume configmaps as environment variable.We created configmaps for spark executors and producers and passed env variables to the deployed pods using configmaps.

With this we were able to remove the hard coded configuration in spark containers and made the containers configurable to be deployed to any cluster.

7 Conclusion and Further Research

We were able to successfully integrate all the components into the cluster and automate the whole process. There are several things that needs to be improved in the future work. They are:

- **Autoscaling:** As soon as the load increases there are ways of horizontal scaling like increase the number of running pods and reducing when not needed and also vertical scaling like adding more resources such as CPU or memory to existing machine.
- **Multiple master nodes** Currently, we have one single master node. If this node goes down, the entire cluster goes down. To avoid this, we need to deploy multiple master nodes

8 References

- Official Kubernetes Documentation
- Ansible Playbook Documentation
- ConfigMap