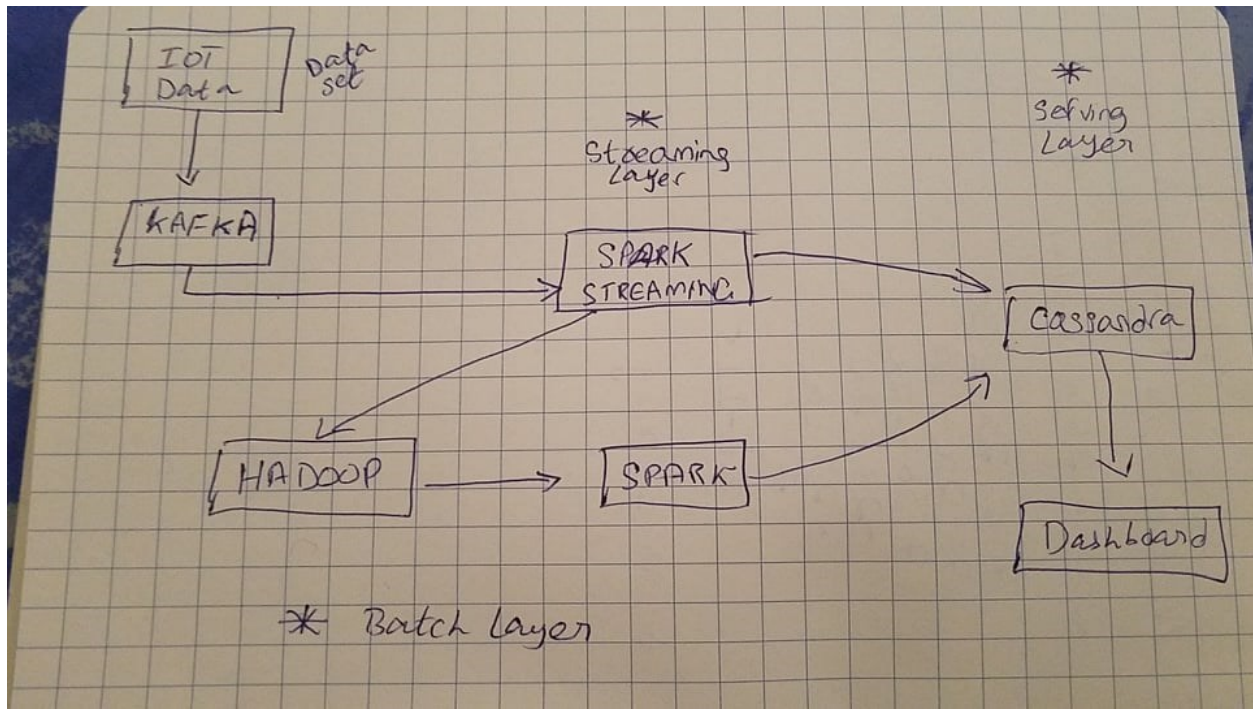


# HADOOP LAB ASSIGNMENT

## ARCHITECTURE:



## DATASET & CODE:

I have used IOT data for this project (below are the sources which I took help from). It generate iot data of a vehicle and transmit its data to KAFKA which then further processed and send to other things.

[https://github.com/uzairaj/lambda\\_arc](https://github.com/uzairaj/lambda_arc)

Sources:

<https://www.infoq.com/articles/traffic-data-monitoring-iot-kafka-and-spark-streaming/>

## STEPS:

For docker-compose

Package initialize

- 1) mvn package

Create a network

2) `sudo docker network create -d bridge lambda_net`

```
uzair@uzair-VirtualBox:~/arch$ docker network create -d bridge lambda_net
WARNING: Error loading config file: /home/uzair/.docker/config.json: stat /home/uzair/.docker/config.json: permission denied
F3af5d0aab930cb27e489f73d2ae15a01a4e32c60eab19f66c40caea9548b8e2
```

docker-compose command, start all containers in one go

3) `sudo docker-compose up -d`

```
Creating zookeeper      ... done
Creating namenode       ... done
Creating cassandra      ... done
Creating spark-master   ... done
Creating spark-worker   ... done
Creating kafka          ... done
Creating datanode       ... done
Creating arch_kafka-manager_1 ... done
```

4) `sudo docker ps # for listing container/images`

| CONTAINER ID | IMAGE                                         | COMMAND                  | CREATED       | STATUS                          |
|--------------|-----------------------------------------------|--------------------------|---------------|---------------------------------|
| f9e84f76b100 | sheepkiller/kafka-manager:latest              | "/start-kafka-manag..."  | 2 minutes ago | Up 2 minutes                    |
| 8f21fd6a07d8 | bde2020/hadoop-datanode:1.1.0-hadoop2.8-java8 | "/entrypoint.sh /run..." | 3 minutes ago | Up 2 minutes (health: starting) |
| ad66edb8b56a | confluentinc/cp-kafka:5.1.0                   | "/etc/confluent/dock..." | 3 minutes ago | Up 2 minutes                    |
| 9d25d6b59d80 | bde2020/spark-worker:2.4.0-hadoop2.7          | "/bin/bash /worker.sh"   | 3 minutes ago | Up 2 minutes                    |
| f0b515826e83 | bde2020/hadoop-namenode:1.1.0-hadoop2.8-java8 | "/entrypoint.sh /run..." | 3 minutes ago | Up 3 minutes (health: starting) |
| 323ef249e934 | bde2020/spark-master:2.4.0-hadoop2.7          | "/bin/bash /master.sh"   | 3 minutes ago | Up 3 minutes                    |
| d8b05947df9c | bitnami/cassandra:latest                      | "/opt/bitnami/script..." | 3 minutes ago | Up 3 minutes                    |
| 30ba59c5ba0f | confluentinc/cp-zookeeper:5.1.0               | "/etc/confluent/dock..." | 3 minutes ago | Up 3 minutes                    |
|              | 2888/tcp, 0.0.0.0:2181->2181/tcp, 3888/tcp    |                          |               | zookeeper                       |

## CREATE A SWARM:

Before creating a swarm we have to find why we need a swarm of docker, what is the the difference between docker compose and docker swarm

***Docker Swarm runs multi-container applications, just like Compose does. Difference is that Swarm schedules and manages your containers across multiple machines, while Compose schedules and manages containers on a single host only.***

## Swarm Steps:

### Initialize swarm

- 1) `sudo docker swarm init`

```
uzair@uzair-VirtualBox:~/arch$ sudo docker swarm init
Swarm initialized: current node (ftcz4kqi5ct6ljxaio4e79uwv) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-5x54m3skdein9kx14nmhbtwnk5tffyia8se8ian0diensuzm9p-68um18owlv4d2tcnoxvxyzr4 10.0.2.15:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
```

Then we use deploy command to stack composer file on top of swarm

Docker stack is used to deploy a complete application stack to the swarm. The deploy command accepts a stack description in the form of a Compose file (yaml file).

- 2) `sudo docker stack deploy --compose-file docker-compose.yml nodes_stack`

```
Creating service nodes_stack_kafka-manager
Creating service nodes_stack_cassandra
Creating service nodes_stack_spark-master
Creating service nodes_stack_spark-worker-1
Creating service nodes_stack_namenode
Creating service nodes_stack_datanode
Creating service nodes_stack_zookeeper
Creating service nodes_stack_kafka
```

- 3) `sudo docker service ls # printing service list`

```
uzair@uzair-VirtualBox:~/arch$ sudo docker service ls
[sudo] password for uzair:
ID                NAME                                MODE                REPLICAS            IMAGE
PORTS
c5k8dx188494     nodes_stack_cassandra             replicated          1/1                 bitnami/cassandra:latest
*:9042->9042/tcp
lijrphff3r5s     nodes_stack_datanode             replicated          1/1                 bde2020/hadoop-datanode:1.1.0-hadoop2.8-java8
*:50010->50010/tcp, *:50075->50075/tcp
w648ibiul7nk     nodes_stack_kafka                 replicated          1/1                 confluentinc/cp-kafka:5.1.0
*:9092->9092/tcp
6lnbswltgx4j     nodes_stack_kafka-manager         replicated          1/1                 sheepkiller/kafka-manager:latest
*:9000->9000/tcp
r2q8bsvtmeok     nodes_stack_namenode             replicated          1/1                 bde2020/hadoop-namenode:1.1.0-hadoop2.8-java8
*:8020->8020/tcp, *:50070->50070/tcp
xf21tww1c2qc     nodes_stack_spark-master          replicated          1/1                 bde2020/spark-master:2.4.0-hadoop2.7
*:4040-4041->4040-4041/tcp, *:7077->7077/tcp, *:8080->8080/tcp
ssb2cdr68isp     nodes_stack_spark-worker-1        replicated          1/1                 bde2020/spark-worker:2.4.0-hadoop2.7
*:8081->8081/tcp
92h2wusl3zfz     nodes_stack_zookeeper             replicated          1/1                 confluentinc/cp-zookeeper:5.1.0
*:2181->2181/tcp
```

- 4) `sudo docker info`

```
Swarm: active
NodeID: ftcz4kqi5ct6ljxaio4e79uwv
Is Manager: true
ClusterID: kpt2eppqib6aatraljtodq3mm
Managers: 1
Nodes: 1
Default Address Pool: 10.0.0.0/16
```

For creating topic and executing sql file which will create tables in Cassandra and create directories of hdfs

5) `./project-orchestrate.sh`

Java command that will print iot data on terminal that will be read by KAFKA consumer or push to Spark streaming as streaming data

6) `java -jar iot-kafka-producer/target/iot-kafka-producer-1.0.0.jar`

Insert data into spark to Cassandra and hdfs \*\* not working facing connection issue due to this data couldn't be added in Cassandra db

### **Kafka:**

Change topicname in Kafka properties in Iot Kafka folder

Create topic: `kafka-topics --create --topic topicname --partitions 1 --replication-factor 1 --if-not-exists --zookeeper zookeeper:2181`

List Topic: `kafka-topics --list --zookeeper zookeeper:2181`

Consumer: `kafka-console-producer --request-required-acks 1 --broker-list kafka:9092 --topic topicname`

### **Cassandra DB:**

`sudo docker exec -it Cassandra bash`

`cqlsh --username cassandra --password Cassandra`

`use TrafficKeySpace;`

`SELECT * FROM TrafficKeySpace.Total_Traffic;`

### **PRINTING DATA**

7) `java -jar iot-springboot-dashboard/target/iot-springboot-dashboard-1.0.0.jar`

## SPARK:

For transferring data from spark to hdfs and cassandra

- 8) `sudo docker exec spark-master /spark/bin/spark-submit --class com.iot.app.spark.processor.StreamingProcessor --master spark://localhost:7077 /opt/spark-data/iot-spark-processor-1.0.0.jar`

For sending hdfs data to spark

- 9) `sudo docker exec spark-master /spark/bin/spark-submit --class com.iot.app.spark.processor.BatchProcessor --master spark://localhost:7077 /opt/spark-data/iot-spark-processor-1.0.0.jar`



**Spark Master at spark://spark-master:7077**

URL: spark://spark-master:7077  
Alive Workers: 1  
Cores in use: 3 Total, 3 Used  
Memory in use: 4.3 GB Total, 1024.0 MB Used  
Applications: 1 Running, 0 Completed  
Drivers: 0 Running, 0 Completed  
Status: ALIVE

▼ Workers (1)

| Worker Id                              | Address          | State | Cores      | Memory |
|----------------------------------------|------------------|-------|------------|--------|
| worker-20190406130351-172.22.0.8-40139 | 172.22.0.8:40139 | ALIVE | 3 (3 Used) | 4.3 GB |

Spark Master

<http://localhost:8080>

Spark Worker Node

<http://localhost:8081>

## HDFS:

<http://localhost:50071>

\*\*\*\*\* Unfortunately due to connection issues wouldn't be able to add image of output results

As I mentioned in last class I had no background in docker, but after this and previous assignments I had some command now and learn a lot of things related to it.

I had gave my best to this project but wouldn't be able to complete whole project due to connection issues between containers in the end. Also I took reference from multiple sources and use built in dashboard \*\*\*\*\*

## **References:**

<https://rominirani.com/docker-swarm-tutorial-b67470cf8872>

<https://www.educba.com/docker-swarm-architecture/>

[https://www.youtube.com/watch?v=2z6scTH\\_C4c](https://www.youtube.com/watch?v=2z6scTH_C4c).

<https://github.com/EthicalML/kafka-spark-streaming-zeppelin-docker/blob/master/docker-compose.yml>

<https://www.youtube.com/watch?v=yrOm5koeKzc>

<https://hands-on-tech.github.io/2018/11/01/kafka-spark-flink-example.html>

<https://www.baeldung.com/kafka-spark-data-pipeline>

<https://medium.com/rahasak/end-to-end-streaming-from-kafka-to-cassandra-447d0e6ba25a>

[https://www.instaclustr.com/wp-content/uploads/2017/09/Using-Apache-Spark-Kafka-and-Cassandra-to-power-intelligent-application\\_final.pdf](https://www.instaclustr.com/wp-content/uploads/2017/09/Using-Apache-Spark-Kafka-and-Cassandra-to-power-intelligent-application_final.pdf)

<https://docs.docker.com/engine/swarm/stack-deploy/>

<https://medium.com/@michaeljpr/five-minute-guide-getting-started-with-cassandra-on-docker-4ef69c710d84>

<https://www.infoq.com/articles/traffic-data-monitoring-iot-kafka-and-spark-streaming/>

THANK YOU 😊