

01 HDFS Practice Environment -KirkYagami

What is a Hadoop Cluster?

A Hadoop cluster is a collection of computers, known as nodes, that are networked together to perform parallel computations on big data sets. The Name node is the master node of the Hadoop Distributed File System (HDFS). It maintains the meta data of the files in the RAM for quick access. An actual Hadoop Cluster setup involves extensive resources which are not within the scope of this lab. In this lab, you will use dockerized hadoop to create a Hadoop Cluster which will have:

1. Namenode
2. Datanode
3. Node Manager
4. Resource manager
5. Hadoop history server
6. git clone

Execute the commands as it as.

```
git clone https://github.com/KirkYagami/docker_hadoop.git
cd docker_hadoop
```

```
docker-compose up -d
```

Compose is a tool for defining and running multi-container Docker applications. It uses the YAML file to configure the services and enables us to create and start all the services from just one configuration file.

```
# 3192219afd04 Pull complete
# aa53513fe997 Pull complete
# b0d764123f3e Pull complete
# b04394ddb35d Pull complete
[+] Running 9/9
# Network ooxwv-docker_hadoop_default Created
# Volume "ooxwv-docker_hadoop_hadoop_historyserver" Created
# Volume "ooxwv-docker_hadoop_hadoop_namenode" Created
# Volume "ooxwv-docker_hadoop_hadoop_datanode" Created
# Container nodemanager Started
# Container datanode Started
# Container historyserver Started
# Container namenode Started
# Container resourcemanager Started
```

To Run HDFS Commands

Start your docker engine and check whether the docker-hadoop container is running or not.

Run the namenode as a mounted drive on bash.

```
docker exec -it namenode /bin/bash
```

Explore the hadoop environment

As you have learnt in the videos and reading thus far in the course, a Hadoop environment is configured by editing a set of configuration files:

- **hadoop-env.sh** Serves as a master file to configure YARN, HDFS, MapReduce, and Hadoop-related project settings.
- **core-site.xml** Defines HDFS and Hadoop core properties
- **hdfs-site.xml** Governs the location for storing node metadata, fsimage file and log file.
- **mapred-site.xml** Lists the parameters for MapReduce configuration.
- **yarn-site.xml** Defines settings relevant to YARN. It contains configurations for the Node Manager, Resource Manager, Containers, and Application Master.

For the docker image, these xml files have been configured already. You can see these in the directory **/opt/hadoop-3.2.1/etc/hadoop/** by running

```
ls /opt/hadoop-3.2.1/etc/hadoop/*.xml  
ls -ltr /opt/hadoop-3.2.1/etc/hadoop/*.xml
```

```
PowerShell 7.4.4
PS C:\Users\NikhilSharma> docker exec -it namenode /bin/bash
root@62cad9af6c4b:/# ls /opt/hadoop-3.2.1/etc/hadoop/*.xml
/opt/hadoop-3.2.1/etc/hadoop/capacity-scheduler.xml /opt/hadoop-3.2.1/etc/hadoop/kms-acls.xml
/opt/hadoop-3.2.1/etc/hadoop/core-site.xml /opt/hadoop-3.2.1/etc/hadoop/kms-site.xml
/opt/hadoop-3.2.1/etc/hadoop/hadoop-policy.xml /opt/hadoop-3.2.1/etc/hadoop/mapred-site.xml
/opt/hadoop-3.2.1/etc/hadoop/hdfs-site.xml /opt/hadoop-3.2.1/etc/hadoop/yarn-site.xml
/opt/hadoop-3.2.1/etc/hadoop/httpfs-site.xml
root@62cad9af6c4b:/# ls -ltr /opt/hadoop-3.2.1/etc/hadoop/*.xml
-rw-r--r-- 1 1001 1001 11392 Sep 10 2019 /opt/hadoop-3.2.1/etc/hadoop/hadoop-policy.xml
-rw-r--r-- 1 1001 1001 682 Sep 10 2019 /opt/hadoop-3.2.1/etc/hadoop/kms-site.xml
-rw-r--r-- 1 1001 1001 3518 Sep 10 2019 /opt/hadoop-3.2.1/etc/hadoop/kms-acls.xml
-rw-r--r-- 1 1001 1001 620 Sep 10 2019 /opt/hadoop-3.2.1/etc/hadoop/httpfs-site.xml
-rw-r--r-- 1 1001 1001 8260 Sep 10 2019 /opt/hadoop-3.2.1/etc/hadoop/capacity-scheduler.xml
-rw-r--r-- 1 1001 1001 2924 Aug 13 18:58 /opt/hadoop-3.2.1/etc/hadoop/core-site.xml
-rw-r--r-- 1 1001 1001 5115 Aug 13 18:58 /opt/hadoop-3.2.1/etc/hadoop/hdfs-site.xml
-rw-r--r-- 1 1001 1001 14660 Aug 13 18:58 /opt/hadoop-3.2.1/etc/hadoop/yarn-site.xml
-rw-r--r-- 1 1001 1001 5203 Aug 13 18:58 /opt/hadoop-3.2.1/etc/hadoop/mapred-site.xml
root@62cad9af6c4b:/# cd /opt/hadoop-3.2.1/etc/hadoop/
root@62cad9af6c4b:/opt/hadoop-3.2.1/etc/hadoop# ls
capacity-scheduler.xml      hadoop-user-functions.sh.example  kms-log4j.properties          ssl-client.xml.example
configuration.xsl          hdfs-site.xml                    kms-site.xml                  ssl-server.xml.example
container-executor.cfg     httpfs-env.sh                    log4j.properties             user_ec_policies.xml.template
core-site.xml              httpfs-log4j.properties          mapred-env.cmd               workers
hadoop-env.cmd             httpfs-signature.secret          mapred-env.sh                yarn-env.cmd
hadoop-env.sh              httpfs-site.xml                  mapred-queues.xml.template   yarn-env.sh
hadoop-metrics2.properties kms-acls.xml                      mapred-site.xml              yarn-site.xml
hadoop-policy.xml          kms-env.sh                       shellprofile.d                yarnservice-log4j.properties
root@62cad9af6c4b:/opt/hadoop-3.2.1/etc/hadoop# vi core-site.xml
```

```
Put site-specific property overrides in this file.

<configuration>
  <property><name>hadoop.proxyuser.hue.hosts</name><value>*</value></property>
  <property><name>fs.defaultFS</name><value>hdfs://namenode:9000</value></property>
  <property><name>hadoop.http.staticuser.user</name><value>root</value></property>
  <property><name>io.compression.codecs</name><value>org.apache.hadoop.io.compress.SnappyCodec</value></property>
  <property><name>hadoop.proxyuser.hue.groups</name><value>*</value></property>
  <property><name>hadoop.proxyuser.hue.hosts</name><value>*</value></property>
  <property><name>fs.defaultFS</name><value>hdfs://namenode:9000</value></property>
  <property><name>hadoop.http.staticuser.user</name><value>root</value></property>
  <property><name>io.compression.codecs</name><value>org.apache.hadoop.io.compress.SnappyCodec</value></property>
  <property><name>hadoop.proxyuser.hue.groups</name><value>*</value></property>
  <property><name>hadoop.proxyuser.hue.hosts</name><value>*</value></property>
  <property><name>fs.defaultFS</name><value>hdfs://namenode:9000</value></property>
  <property><name>hadoop.http.staticuser.user</name><value>root</value></property>
  <property><name>io.compression.codecs</name><value>org.apache.hadoop.io.compress.SnappyCodec</value></property>
  <property><name>hadoop.proxyuser.hue.groups</name><value>*</value></property>
  <property><name>hadoop.proxyuser.hue.hosts</name><value>*</value></property>
  <property><name>fs.defaultFS</name><value>hdfs://namenode:9000</value></property>
  <property><name>hadoop.http.staticuser.user</name><value>root</value></property>
  <property><name>io.compression.codecs</name><value>org.apache.hadoop.io.compress.SnappyCodec</value></property>
  <property><name>hadoop.proxyuser.hue.groups</name><value>*</value></property>
</configuration>
root@62cad9af6c4b:/opt/hadoop-3.2.1/etc/hadoop#
```

1. In the HDFS, create a directory structure named `user/root/input`.

```
hdfs dfs -mkdir -p /user/root/input
```

2. Copy all the hadoop configuration xml files into the input directory.

```
hdfs dfs -put $HADOOP_HOME/etc/hadoop/*.xml /user/root/input
```

3. Create a `data.txt` file in the current directory.

```
curl  
https://raw.githubusercontent.com/KirkYagami/docker_hadoop/master/SampleMapReduce.t  
xt --output data.txt
```

4. Copy the `data.txt` file into `/user/root`.

```
hdfs dfs -put data.txt /user/root/
```

5. Check if the file has been copied into the HDFS by viewing its content.

```
hdfs dfs -cat /user/root/data.txt
```