Building a Hadoop Cluster

ALAOUI Brahim

Computer Science Department, University of Chouaib Doukkali, Eljadida, Morrocco.

Contributing authors: brahimalaoui0216@gmail.com

Abstract

This article explores the process of building a cluster using Apache Hadoop, a popular open-source framework for handling large-scale data processing. The increasing volume and complexity of data in modern applications require scalable and distributed solutions, making Hadoop an ideal choice. The article covers the key steps in setting up a Hadoop cluster, including infrastructure requirements, cluster management tools, and deployment strategies. It highlights the benefits of Hadoop's distributed storage (HDFS) and processing capabilities (MapReduce), fault tolerance mechanisms, and support for various programming languages. Additionally, the article discusses cluster optimization techniques such as data partitioning, caching, and data compression to enhance performance and resource utilization. It also addresses cluster monitoring and debugging tools to ensure efficient operation. By providing insights into the cluster construction process, this article enables harnessing the power of Hadoop and building robust and high-performance data processing environments.

Keywords: Cluster, Apache Hadoop, Big Data, Data processing

1. Introduction

In the era of big data, where information flows ceaselessly from diverse sources, the ability to process, analyze, and derive insights from massive datasets has become paramount. Apache Hadoop, an open-source, robust, and scalable big data processing framework, has become a critical solution for organizations seeking to tackle these data challenges head-on. Its core components, the Hadoop Distributed File System (HDFS) for storage and MapReduce for processing, have transformed the way we approach data analytics, machine learning, and more.

However, harnessing the full potential of Apache Hadoop requires more than just software installation and a few lines of code. To truly unlock its power and efficiency, one must construct a well-optimized Hadoop cluster—a network of interconnected computers that work in harmony to manage and process vast datasets in parallel. This article serves as your comprehensive guide to building a high-performance Hadoop cluster from the ground up. We will embark on a journey through the intricacies of planning, hardware selection, configuration, and maintenance, providing you with the knowledge and expertise needed to construct a Hadoop cluster tailored to your specific data processing needs.

2. Understanding Apache Hadoop

Apache Hadoop is an open-source cluster computing framework for large-scale data processing. The main feature of Apache Hadoop is its distributed storage and processing capabilities, which enable efficient handling of vast datasets. Hadoop provides an interface for programming entire clusters with implicit data parallelism and fault tolerance. It is designed to cover a wide range of workloads, including batch processing, iterative algorithms, and interactive queries, but it excels particularly in handling large, unstructured data through its Hadoop Distributed File System (HDFS) and MapReduce processing model.

Fig. 1 HADOOP's features.

- **Distributed File System**: The Hadoop Distributed File System (HDFS) is designed to store large datasets reliably and to stream those datasets at high bandwidth to user applications. It splits data into blocks and distributes them across multiple nodes in the cluster.
- **Ecosystem Approach**: Hadoop has a comprehensive ecosystem that includes various tools and frameworks such as Apache Hive, Apache Pig, Apache HBase, Apache Spark, and Apache Flink. These tools extend Hadoop's capabilities for different types of data processing and analysis tasks, making it versatile and powerful.
- **Big Data Analytics**: Hadoop is optimized for processing and analyzing large datasets. It allows for the execution of complex data queries and analytics through distributed computing, making it suitable for big data analytics applications.
- **Open Source:** Hadoop is an open-source framework, which means it is freely available for anyone to use, modify, and distribute. This encourages innovation and collaboration within the community and reduces the cost of implementation.
- **Commodity Hardware**: Hadoop is designed to run on commodity hardware, which are inexpensive, standard machines that are widely available. This makes Hadoop a cost-effective solution for large-scale data storage and processing.
- **Flexible**: Hadoop can store and process various types of data, including structured, semi-structured, and unstructured data. This flexibility allows organizations to analyze data from multiple sources and formats.
- **Scalable**: Hadoop clusters can be easily scaled by adding more nodes. This horizontal scalability allows Hadoop to handle increasing volumes of data without significant changes to the existing infrastructure.
- **Reliable**: Hadoop ensures data reliability through data replication. Each block of data is replicated across multiple nodes in the cluster, so if one node fails, the data can still be accessed from another node. This fault tolerance is a key feature of Hadoop.

3. Hadoop Architecture

Hadoop, a free framework based on the MapReduce architecture, groups files into large blocks before providing them to the cluster nodes. These are grouped into two main components:

- Master Nodes.
- Slave Nodes.

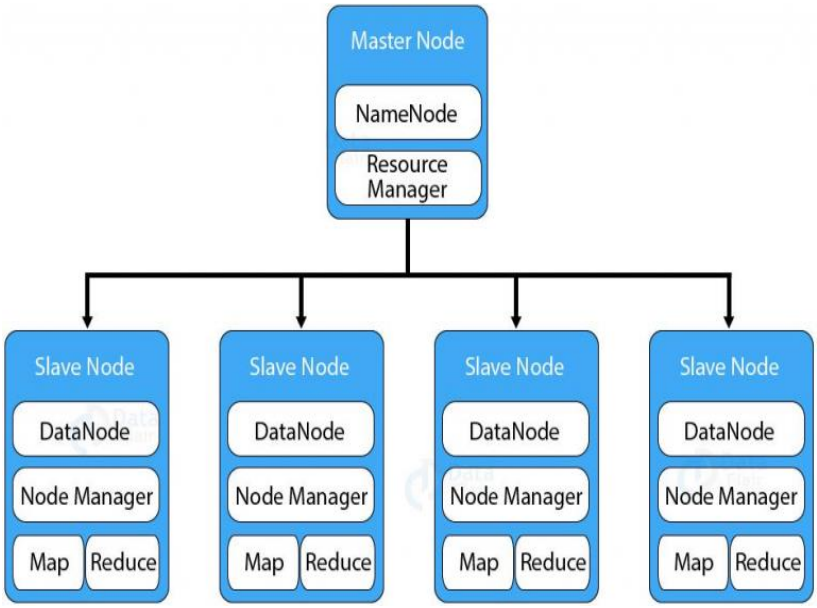


Fig. 2 Hadoop Architecture

The master nodes (Hadoop main nodes) are servers hosting different services such as: storage management or processing.

Slave nodes are machines, which can also be called "worker nodes", which execute the tasks sent by the master nodes. It is in these nodes that the storage and processing of Hadoop data takes place. Several services are therefore used: NodeManager, ApplicationMaster, Container, TaskTracker, DataNode…

4. Setting Up the Hardware

I set up four machines using Linux. These virtual machines were configured to be on the same network, allowing me to connect them to each other. I used the ping tool to verify network connectivity between the machines, and I also configured SSH to establish a secure connection between them. With this configuration, I created a cluster where the virtual machines can work together and share resources. This setup provides an ideal platform for running distributed computing tasks and exploring technologies such as Hadoop, which require a cluster environment.

To create an Apache Hadoop cluster, several steps need to be followed. First, I installed Apache Hadoop on the cluster nodes, whether they are physical machines or cloud instances. Linux is recommended as the operating system for Hadoop development and deployment, although Windows is also supported as a development platform.

Once Hadoop was installed on each cluster node, the master/slave architecture of Hadoop came into play. In this architecture, one node is designated as the master node, responsible for overall task coordination and cluster resource management. The master node typically runs the NameNode service for the Hadoop Distributed File System (HDFS) and the JobTracker service for the MapReduce processing framework.

The other nodes in the cluster are designated as slave nodes. Slave nodes perform the actual tasks of distributed processing and data storage, utilizing the available resources on each machine. They typically run the DataNode service for HDFS and the TaskTracker service for MapReduce.

A resource manager, such as YARN (Yet Another Resource Negotiator), is also part of the Hadoop architecture. YARN facilitates resource management and allocation between the master and slave nodes. It allows for efficient scheduling and execution of distributed data processing jobs.

Once the Hadoop cluster was configured with a master node, slave nodes, and a resource manager, I could submit distributed processing jobs using the Hadoop API. The master node coordinates the distribution of tasks to the slave nodes, which then execute these tasks in parallel on the distributed data within the cluster. This setup allows for efficient and scalable processing of large datasets across multiple nodes.

5. Installing and Configuring Hadoop Cluster

Setting up Apache Hadoop is a crucial step in building your high-performance Hadoop cluster. In this section, we'll guide you through the process of installing and configuring Apache Hadoop on the cluster nodes.

5.1 Change the hostname of all 4 systems

For each Machines you have to specify the hostname accordingly.

```
$ sudo vim /etc/hostname
```

- Master

```
master
```

- Node1 and Node2 and Node3

```
node(1, 2 or 3)
```

Press i on the keyboard and write 'master' by deleting Ubuntu.
Press ESC on the keyboard
Save the configuration by :wq

5.2 Update the hosts on all 4 nodes

Find the ip Address of all 3 systems and try to ping each other.

```
$ ifconfig
```

Change the hosts file in /etc/hosts for Master and all the slaves(1-3):

```
$ sudo vim /etc/hosts
```

Restart all Machines in order to reflect the changes.

5.3 Ping Each other using Hostname

```
$ ping “hostname for all machines (master and nodes)”
```

5.4 Test SSH connectivity

Test the SSH connectivity by doing the following. It will ask for yes or no and you should type 'yes'.

Perform SSH master/node1/node2/node3 on each of the node to verify the connectivity.

```
$ ssh node1
$ ssh node 2
$ ssh node 3
```

Repeat this step for each node to verify the connectivity

It will ask for yes or no and you should type 'yes' We should be able to SSH master and SSH nodes without password prompt. If it asks for password while connecting to master or slave using SSH, there is something went wrong and you need to fix it before proceeding further.

5.5 Update core-site.xml(Master+ All Nodes)

```
$ sudo vim $HADOOP_HOME/etc/Hadoop/core-site.xml
```

The changes you need to make to core-site.xml

Change localhost to master

```
<configuration>
        <property>
           <name>fs.defaultFS</name>
           <value>hdfs://master:9000</value>
        </property>
</configuration>
```

5.6 Update hdfs-site.xml(Master + All Nodes)

```
$ sudo vim $HADOOP_HOME/etc/Hadoop/hdfs-site.xml
```

The changes you need to make to hdfs-site.xml

a. Replication is set to 3

b. Namenode configured only in master

c. Datanode configured only in nodes

- **Master only**

```
<configuration>
        <property>
                <name>dfs.replication</name>
                <value>3</value>
        </property>
        <property>
                <name>dfs.namenode.name.dir</name>
                <value>/home/brahim/data/hadoop_temp/hdfs/namenode</value>
        </property>
</configuration>
```

- **Node1/Node2/Node3 only**

```
<configuration>
        <property>
                <name>dfs.replication</name>
                <value>3</value>
        </property>
        <property>
                <name>dfs.namenode.name.dir</name>
                <value>/home/brahim/data/hadoop_temp/hdfs/datanode</value>
        </property>
</configuration>
```

5.7 Update yarn-site.xml(Master + All Nodes)

```
$ sudo vim $HADOOP_HOME/etc/Hadoop/yarn-site.xml
```

```
<configuration>
        <property>
                <name>yarn.nodemanager.aux-services</name>
                <value>mapreduce_shuffle</value>
        </property>
        <property>
                <name>yarn.resourcemanager.resource-tracker.address</name>
                <value>master:8025</value>
        </property>
        <property>
                <name>yarn.resourcemanager.scheduler.address</name>
                <value>master: 8030</value>
        </property>
        <property>
                <name>yarn.resourcemanager.address</name>
                <value>master:8050</value>
        </property>
 </configuration>
```

5.8 Update mapred-site.xml(Master + All Nodes)

```
$ sudo vim $HADOOP_HOME/etc/Hadoop/mapred-site.xml
```

```
<configuration>

        <property>

                <name>mapreduce.framework.name</name>

                <value>yarn</value>

        </property>

        <property>

                <name>mapreduce.jobhistory.address</name>

                <value>master:10020</value>

        </property>

 </configuration>
```

6. Master Configuration

Update Masters and slaves file (Master Node only) If you see any entry related to localhost, feel free to delete it. This file is just helper file that are used by Hadoop scripts to start appropriate services on master and nodes.

```
$ vim $HADOOP_HOME/etc/Hadoop/slaves
```

- Edit Slaves file:

```
node1
node2
node3
```

- Below masters file does not exists by default. It gets created the files:

```
$ vim $HADOOP_HOME/etc/Hadoop/masters
```

```
master
```

Note: You don't need to configure them in slave nodes

6.1 Recreate Namenode folder

First you have to delete the older hdfs file.

```
$ sudo rm -rf /home/brahim/Desktop/hdfs
```

Then create another file that contains hdfs file like.

```
$ sudo rm -rf /home/brahim/data/Hadoop_tmp/hdfs/namenode
```

Then type these commands:

```
$ sudo chown brahim:Hadoop -R /home/brahim/data/Hadoop_tmp/
```

```
$ sudo chmod 777 /home/brahim/data/Hadoop_tmp/hdfs/namenode
```

6.2 Change the path in hdfs-site.xml

```
<configuration>
        <property>
                <name>dfs.replication</name>
                <value>3</value>
        </property>
        <property>
                <name>dfs.namenode.name.dir</name>
                <value>/home/brahim/data/hadoop_temp/hdfs/namenode</value>
        </property>
</configuration>
```

7. Slave Configuration (All Nodes Only)

First you have to delete the older hdfs file.

```
$ sudo rm -rf /home/brahim/Desktop/hdfs
```

Then create another file that contains hdfs file like.

```
$ sudo rm -rf /home/brahim/data/Hadoop_tmp/hdfs/datanode
```

Then type these commands:

```
$ sudo chown brahim:Hadoop -R /home/brahim/data/Hadoop_tmp/
```

```
$ sudo chmod 777 /home/brahim/data/Hadoop_tmp/hdfs/datanode
```

7.1 Change the path in hdfs-site.xml

```
<configuration>
        <property>
                <name>dfs.replication</name>
                <value>3</value>
        </property>
        <property>
                <name>dfs.namenode.name.dir</name>
                <value>/home/brahim/data/hadoop_temp/hdfs/datanode</value>
        </property>
</configuration>
```

8. Format the Namenode(Master only)

Before starting the cluster, we need to format the Namenode. Use the following command only on master node:

```
$ hdfs namenode-format
```

9. Start the DFS & Yarn (Master Only)

```
$ start-all.sh
```

You should observe that it tries to start data node on slave nodes one by one. Once it is started, Do a JPS on Master and slaves.

- **JPS on Master node**

```
$ jps

    SecondaryNameNode
    NameNode
    Jps
    ResourceManager
```

- **JPS on slave nodes(node1 et node2 et node3)**

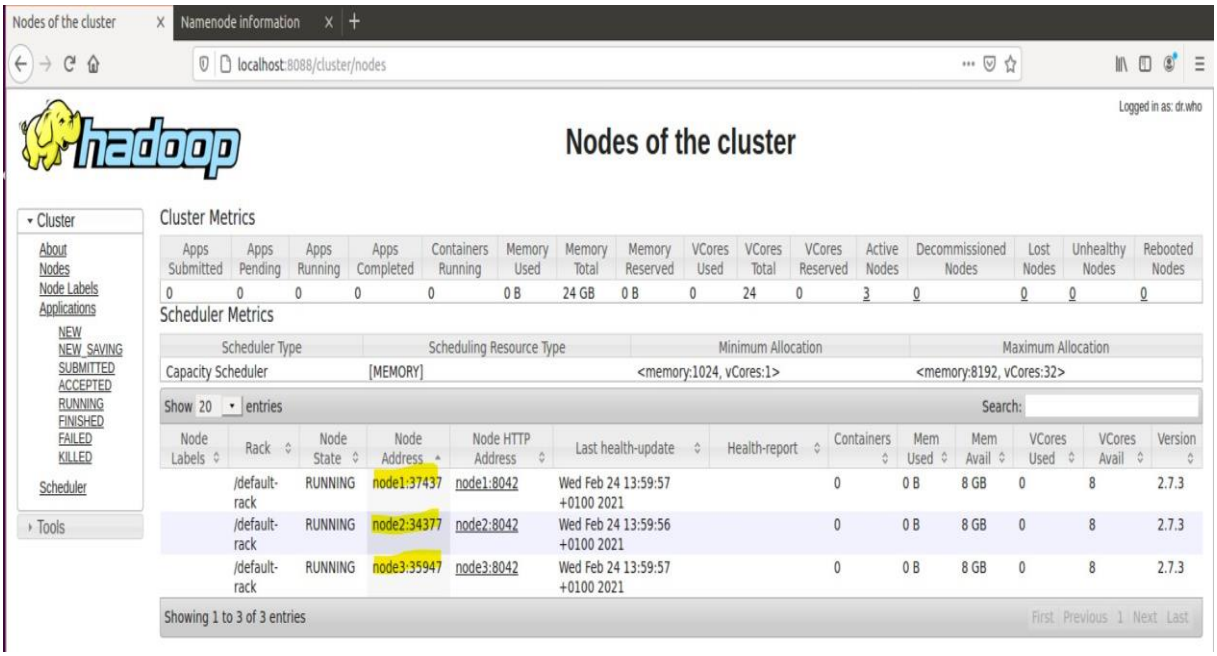
```
$ jps

    Jps
    NodeManager
    DataNode
```

10. Review Yarn console

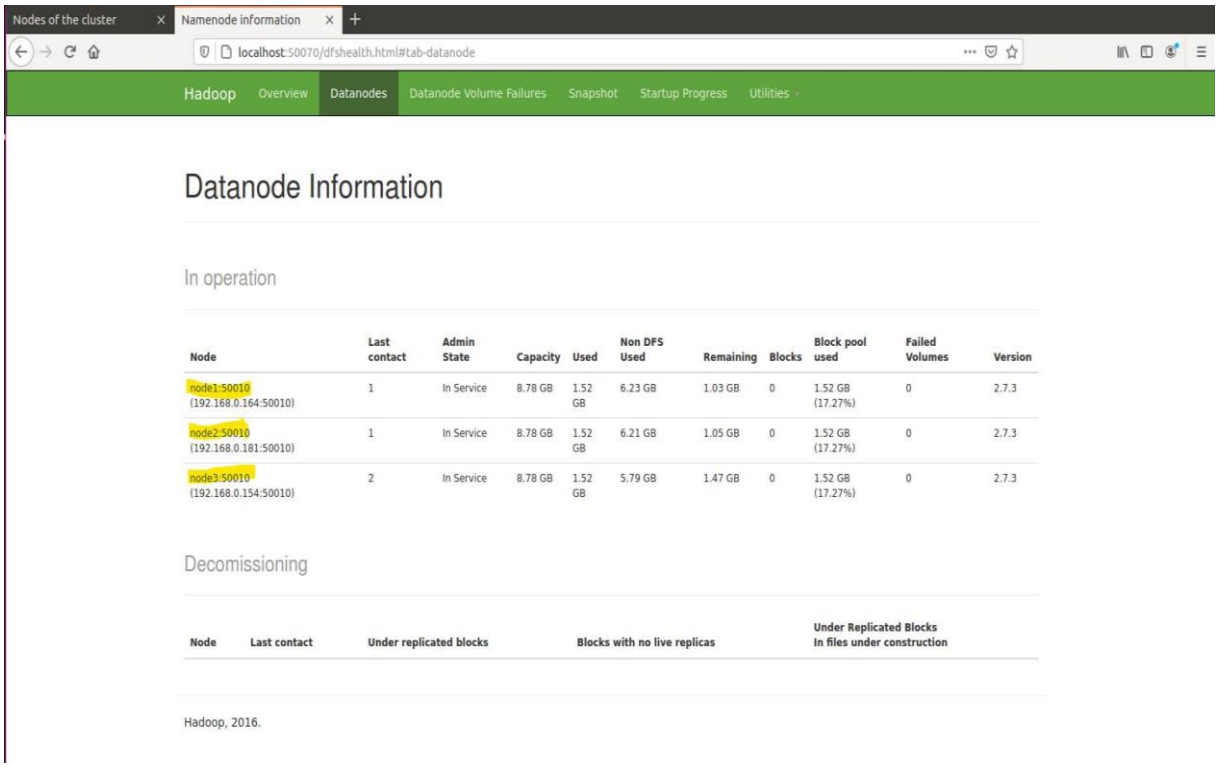
If all the services started successfully on all nodes, then you should see all of your nodes listed under Yarn nodes. You can hit the following url on your browser and verify that:

- http://master:8088/cluster/nodes



- http://master:50070

can show live node count and info about each live node



Conclusion

In the realm of big data processing and analytics, a well-constructed Hadoop cluster serves as the cornerstone for unlocking the full potential of Apache Hadoop. Through the journey outlined in this article, you've been equipped with the knowledge and practical steps required to build a high-performance Hadoop cluster from scratch.

From the initial planning and hardware setup to the intricate configuration and optimization, you've learned that building a Hadoop cluster is not merely a technical endeavor but a strategic investment. This investment opens the doors to a world of distributed computing capabilities, enabling you to conquer vast datasets and execute complex tasks with efficiency and speed.

By setting up your virtual machines or physical nodes, establishing network connectivity, and configuring Hadoop to suit your specific needs, you've laid the foundation for a dynamic data processing environment. You've also explored the pivotal role of cluster managers and the importance of resource allocation in ensuring seamless operations. With your Hadoop cluster in place, you are now poised to explore a myriad of data processing opportunities. Whether you're diving into the world of machine learning, conducting real-time streaming analytics, or performing large-scale batch processing, your cluster provides the computational power and scalability needed to succeed.

As you move forward, remember that the journey doesn't end with the cluster's creation. Regular maintenance, monitoring, and optimization will be essential to keep your Hadoop cluster running smoothly and efficiently.

In closing, the world of big data holds immense potential, and with your Hadoop cluster as a reliable ally, you're ready to embark on a data-driven adventure. This article has equipped you with the foundational knowledge and practical skills to harness the true power of distributed computing, and we encourage you to explore further, innovate, and uncover new insights in your data journey. Building and maintaining a Hadoop cluster is a step toward conquering the challenges of the data-driven world, and your exploration has only just begun.

References

https://hadoop.apache.org/ [20/06/2024]

Architecture technique de Hadoop, rabarisoanaivo https://sysblog.informatique.univ-paris-diderot.fr/2020/03/26/architecture-technique-de-hadoop/

Step by Step Guide for Hadoop 3 Installation for Windows Machine on Huawei Cloud, Hakan GÜVEZ https://medium.com/huawei-developers/step-by-step-guide-for-hadoop-3-installation-for-windows-machine-on-huawei-cloud-816030b5fdb1