Why GitHub? Team Enterprise Explore Marketplace Pricing Search / Sign in Sign up

ChiaraFor96 / hadoop-on-aws Notifications Star 0 Fork 0

Code Issues Pull requests Actions Projects Wiki Security Insights

main hadoop-on-aws / README.md Go to file ···

ChiaraFor96 Update README.md Latest commit 4d7eba4 5 minutes ago History

1 contributor

428 lines (365 sloc) | 14.4 KB Raw Blame

hadoop-on-aws

In this repository I show you how to setup a plain Hadoop cluster, in particular using EC2 machines (that lives in the same subnet with ubuntu). But you con consider it for a plain cluster of on premis machines.

Single machine cluster set up

This part is expired from these tutorials:

- http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/SingleCluster.html
- https://www.edureka.co/blog/install-hadoop-single-node-hadoop-cluster

Start.

- Download hadoop: `wget https://apache.osuosl.org/hadoop/common/stable/hadoop-3.3.1.tar.gz`
- unpack the tar: `tar -xvf hadoop-3.3.1.tar.gz`
- Install *java* `sudo apt-get -y install openjdk-8-jdk-headless`

From now assume that the upacked tar forlder name is HADOOP_FILES

- Change the `.bashrc` file adding these rows:

```
#set hadoop home
export HADOOP_HOME=$HOME/HADOOP_FILES
export HADOOP_CONF_DIR=$HOME/HADOOP_FILES/etc/hadoop
export HADOOP_MAPRED_HOME=$HOME/HADOOP_FILES
export HADOOP_COMMON_HOME=$HOME/HADOOP_FILES
export HADOOP_HDFS_HOME=$HOME/HADOOP_FILES
export YARN_HOME=$HOME/HADOOP_FILES
export PATH=$PATH:$HOME/HADOOP_FILES/bin

# set java home
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export PATH=/usr/lib/jvm/java-8-openjdk-amd64:$PATH
export PDSH_RCMD_TYPE=ssh
```

- Run `source .bashrc` for reload the file
- modify the configurations file that are in `HADOOP_FILES/etc/hadoop/`
  - for the file `core-site.xml` (it contains configuration settings of Hadoop core such as I/O settings that are common to HDFS & MapReduce)

```
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:9000</value>
</property>
</configuration>
```

  - for the file `hdfs-site.xml` (which is the file with configuration settings of HDFS daemons (i.e. NameNode, DataNode, Secondary NameNode). It also includes the replication factor and block size of HDFS.)

```
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.permission</name>
```

```
<value>false</value>
</property>
</configuration>
```

  - for the file `yarn-site.xml` (it contains configuration settings of MapReduce application like number of JVM that can run in parallel, the size of the mapper and the reducer process, CPU cores available for a process, etc.)

    ```
    <configuration>
    <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
    </property>
    <property>
    <name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
    </property>
    </configuration>
    ```

  - for the file `mapred-site.xml` (it ontains configuration settings of ResourceManager and NodeManager like application memory management size, the operation needed on program & algorithm, etc.)

    ```
    <configuration>
    <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
    </property>
    <property>
    <name>yarn.app.mapreduce.am.env</name>
    <value>HADOOP_MAPRED_HOME=/home/ubuntu/HADOOP_FILES</value>
    </property>
    <property>
    <name>mapreduce.map.env</name>
    <value>HADOOP_MAPRED_HOME=/home/ubuntu/HADOOP_FILES</value>
    </property>
    <property>
    <name>mapreduce.reduce.env</name>
    <value>HADOOP_MAPRED_HOME=/home/ubuntu/HADOOP_FILES</value>
    </property>
    </configuration>
    ```

  - for `hadoop-env.sh` add `export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64`

Open ports 9870 and 8088 to the IPs that you want to see web UIs of hadoop

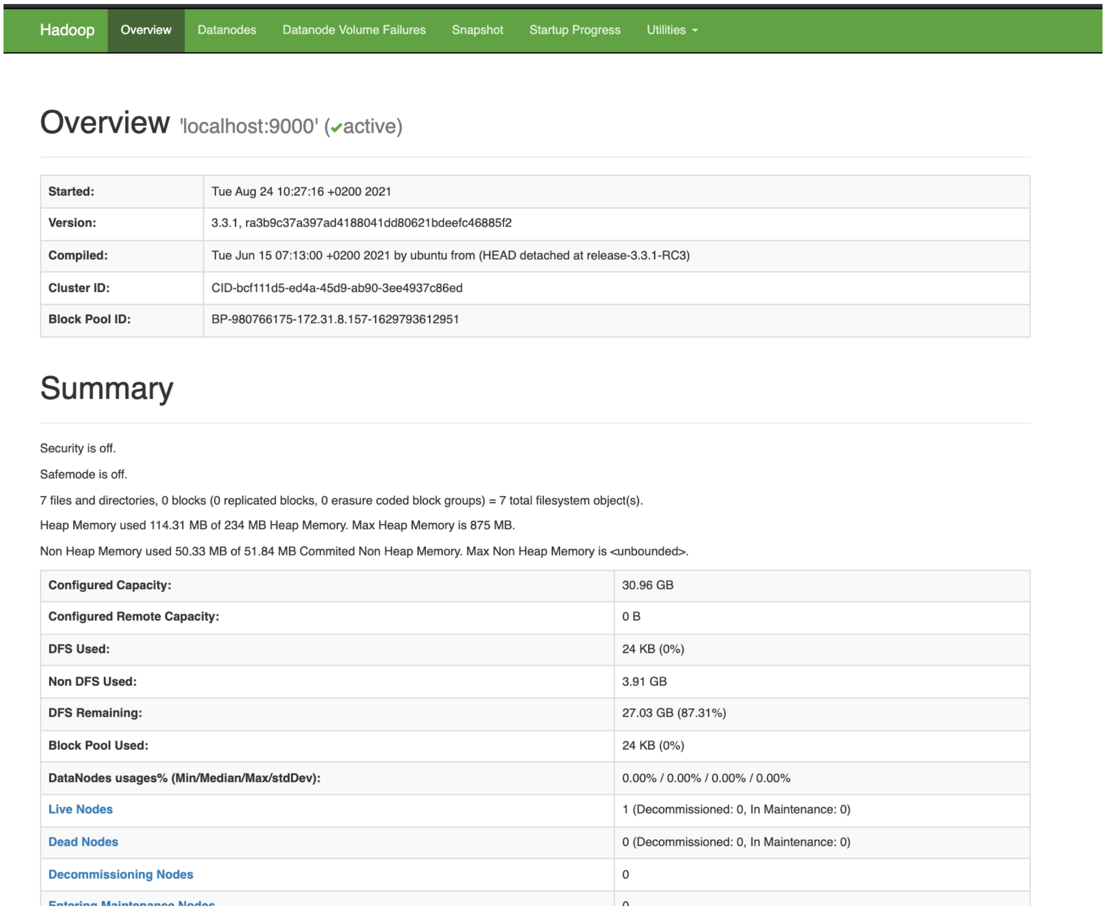
Run the single machine cluster

From the HADOOPFILE FOLDER

- `./hadoop-daemon.sh start namenode` for start name node
- `/hadoop-daemon.sh start datanode` for start the data node
- `./yarn-daemon.sh start resourcemanager` for start the resource manager
- `./yarn-daemon.sh start nodemanager` for start the node manager
- `./mr-jobhistory-daemon.sh start historyserver` for start the history server

Some screens

Overview:



Overview 'localhost:9000' (✓active)

Started:	Tue Aug 24 10:27:16 +0200 2021
Version:	3.3.1, ra3b9c37a397ad4188041dd80621bdeefc46885f2
Compiled:	Tue Jun 15 07:13:00 +0200 2021 by ubuntu from (HEAD detached at release-3.3.1-RC3)
Cluster ID:	CID-bcf111d5-ed4a-45d9-ab90-3ee4937c86ed
Block Pool ID:	BP-980766175-172.31.8.157-1629793612951

Summary

Security is off.

Safemode is off.

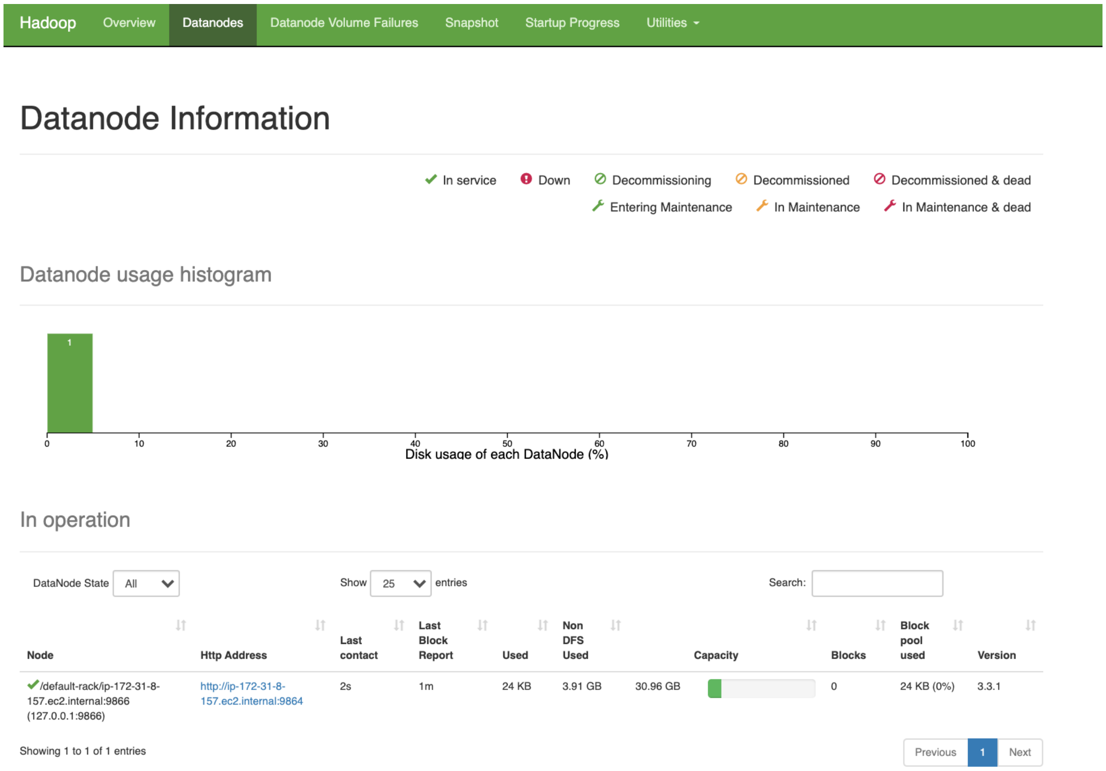
7 files and directories, 0 blocks (0 replicated blocks, 0 erasure coded block groups) = 7 total filesystem object(s).

Heap Memory used 114.31 MB of 234 MB Heap Memory. Max Heap Memory is 875 MB.

Non Heap Memory used 50.33 MB of 51.84 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	30.96 GB
Configured Remote Capacity:	0 B
DFS Used:	24 KB (0%)
Non DFS Used:	3.91 GB
DFS Remaining:	27.03 GB (87.31%)
Block Pool Used:	24 KB (0%)
DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 0.00% / 0.00%
Live Nodes	1 (Decommissioned: 0, In Maintenance: 0)
Dead Nodes	0 (Decommissioned: 0, In Maintenance: 0)
Decommissioning Nodes	0
Entering Maintenance Nodes	0

Cluster Datanode:



Datanode Information

✓ In service ● Down ⊘ Decommissioning ⊘ Decommissioned ⊘ Decommissioned & dead

🔧 Entering Maintenance 🔧 In Maintenance 🔧 In Maintenance & dead

Datanode usage histogram

In operation

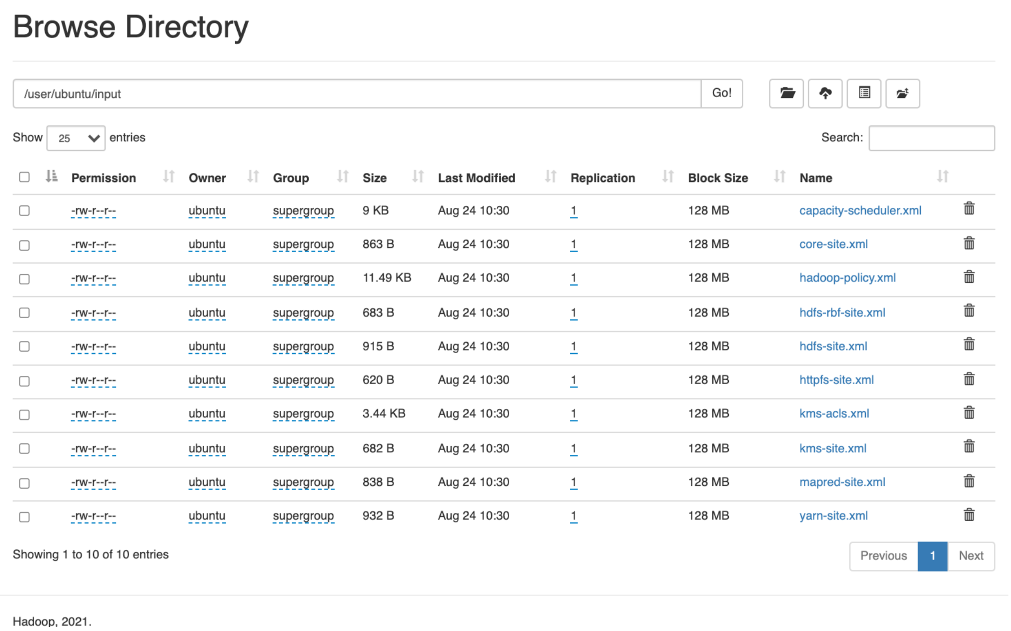
DataNode State All ▾ Show 25 ▾ entries Search:

Node	Http Address	Last contact	Last Block Report	Used	Non DFS Used	Capacity		Blocks	Block pool used	Version
✓/default-rack/ip-172-31-8-157.ec2.internal:9866 (127.0.0.1:9866)	http://ip-172-31-8-157.ec2.internal:9864	2s	1m	24 KB	3.91 GB	30.96 GB		0	24 KB (0%)	3.3.1

Showing 1 to 1 of 1 entries

Previous 1 Next

HDFS interface (http://machineIP:9870/dfshealth.html):



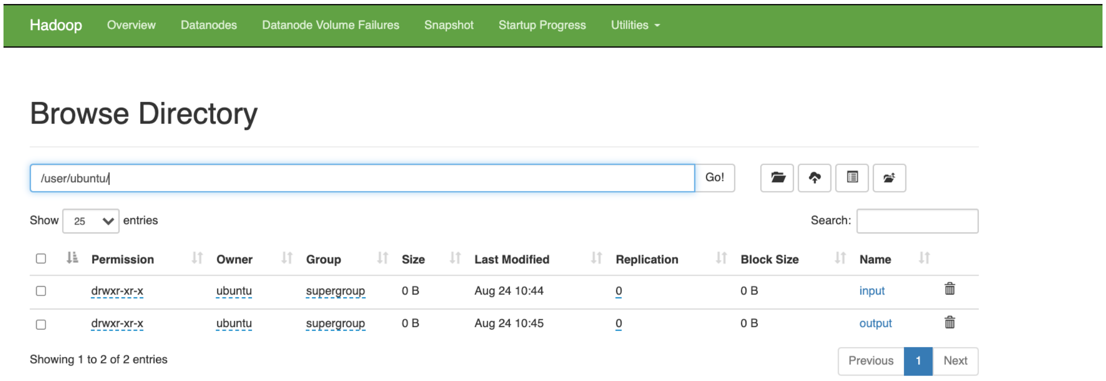
Browse Directory

/user/ubuntu/input

☐	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
☐	-rw-r--r--	ubuntu	supergroup	9 KB	Aug 24 10:30	1	128 MB	capacity-scheduler.xml
☐	-rw-r--r--	ubuntu	supergroup	863 B	Aug 24 10:30	1	128 MB	core-site.xml
☐	-rw-r--r--	ubuntu	supergroup	11.49 KB	Aug 24 10:30	1	128 MB	hadoop-policy.xml
☐	-rw-r--r--	ubuntu	supergroup	683 B	Aug 24 10:30	1	128 MB	hdfs-rbf-site.xml
☐	-rw-r--r--	ubuntu	supergroup	915 B	Aug 24 10:30	1	128 MB	hdfs-site.xml
☐	-rw-r--r--	ubuntu	supergroup	620 B	Aug 24 10:30	1	128 MB	httpfs-site.xml
☐	-rw-r--r--	ubuntu	supergroup	3.44 KB	Aug 24 10:30	1	128 MB	kms-acls.xml
☐	-rw-r--r--	ubuntu	supergroup	682 B	Aug 24 10:30	1	128 MB	kms-site.xml
☐	-rw-r--r--	ubuntu	supergroup	838 B	Aug 24 10:30	1	128 MB	mapred-site.xml
☐	-rw-r--r--	ubuntu	supergroup	932 B	Aug 24 10:30	1	128 MB	yarn-site.xml

Showing 1 to 10 of 10 entries

Hadoop, 2021.



Browse Directory

/user/ubuntu/

☐	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
☐	drwxr-xr-x	ubuntu	supergroup	0 B	Aug 24 10:44	0	0 B	input
☐	drwxr-xr-x	ubuntu	supergroup	0 B	Aug 24 10:45	0	0 B	output

Showing 1 to 2 of 2 entries

Cluster with more than a node set up

For this part I follow this two tutorials and some stackoverflow questions:

- https://www.edureka.co/blog/setting-up-a-multi-node-cluster-in-hadoop-2.X
- http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/ClusterSetup.html

Start.

Download and upack the tar file in each machine, and install some programs:

- Download hadoop: `wget https://apache.osuosl.org/hadoop/common/stable/hadoop-3.3.1.tar.gz`
- unpack the tar: `tar -xvf hadoop-3.3.1.tar.gz`
- Install *java* `sudo apt-get -y install openjdk-8-jdk-headless`
- `sudo apt install firewalld`
- `sudo apt install net-tools`

Disable firewall:

- `sudo systemctl stop firewalld`
- `sudo systemctl disable firewalld`
- `sudo service firewalld stop`
- `sudo ufw disable`

Change `/etc/hosts` file adding private IPs of all cluster machines like follow (in AWS the private ip not changes and the machines are in the same subnet)

```
172.31.4.195 master
172.31.11.53 slave1
172.31.5.5 slave2
```

and other slaves that you have.

From now assume that the upacked tar folder name is HADOOP_FILES

Restart host service: `service sshd restart`

- Change the `.bashrc` file adding these rows:

  ```
  export HADOOP_HOME=$HOME/server/hadoop-3.3.1
  export HADOOP_CONF_DIR=$HOME/HADOOP_FILES/etc/hadoop
  export HADOOP_MAPRED_HOME=$HOME/HADOOP_FILES
  export HADOOP_COMMON_HOME=$HOME/HADOOP_FILES
  export HADOOP_HDFS_HOME=$HOME/HADOOP_FILES
  export HADOOP_YARN_HOME=$HOME/HADOOP_FILES
  export PATH=$PATH:$HOME/HADOOP_FILES/bin

  export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
  export PATH=/usr/lib/jvm/java-8-openjdk-amd64:$PATH
  ```

- Run `source .bashrc` for reload the file

Open ports:

- 9870 and 8088 of the master to the IPs that you want to see web UIs of hadoop

- 0 - 65535 between hosts for communicate each other

- modify the configurations file that are in `HADOOP_FILES/etc/hadoop/` - for the file `core-site.xml` (it contains configuration settings of Hadoop core such as I/O settings that are common to HDFS & MapReduce) for both *master* and *slaves* `<configuration> <property> <name>fs.default.name</name> <value>hdfs://master:9000</value> </property> </configuration>`

  - for the file `hdfs-site.xml` (which is the file with configuration settings of HDFS daemons (i.e. NameNode, DataNode, Secondary NameNode). It also includes the replication factor and block size of HDFS.)

    for *master*

    ```
    <configuration>
    <property>
    <name>dfs.replication</name>
    <value>2</value>
    </property>
    <property>
    <name>dfs.permissions</name>
    <value>false</value>
    </property>
    <property>
    <name>dfs.namenode.name.dir</name>
    <value>/home/ubuntu/server/hadoop-3.3.1/namenode</value>
    </property>
    <property>
    <name>dfs.datanode.data.dir</name>
    <value>/home/ubuntu/server/hadoop-3.3.1/datanode</value>
    </property>
    </configuration>
    ```

    for *slaves*

    ```
    <configuration>
    <property>
    <name>dfs.replication</name>
    <value>2</value>
    </property>
    <property>
    <name>dfs.permissions</name>
    <value>false</value>
    </property>
    <property>
    <name>dfs.datanode.data.dir</name>
    <value>/home/ubuntu/server/HADOOP_FILES/datanode</value>
    </property>
    </configuration>
    ```

    Remember to mkdir of the datanode folder on slaves (if not preset on them) and chmod 700

  - for the file `yarn-site.xml` (it contains configuration settings of MapReduce application like number of JVM that can run in parallel, the size of the mapper and the reducer process, CPU cores available for a process, etc.) for both *master* and *slaves*

    ```
    <configuration>
    <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
    </property>
    <property>
    <name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
    </property>
    </configuration>
    ```

  - for the file `mapred-site.xml` (cit ontains configuration settings of ResourceManager and NodeManager like application memory management size, the operation needed on program & algorithm, etc.) for both *master* and *slaves*

```
<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
<property>
<name>yarn.app.mapreduce.am.env</name>
<value>HADOOP_MAPRED_HOME=/home/ubuntu/HADOOP_FILES</value>
</property>
<property>
<name>mapreduce.map.env</name>
<value>HADOOP_MAPRED_HOME=/home/ubuntu/HADOOP_FILES</value>
</property>
<property>
<name>mapreduce.reduce.env</name>
<value>HADOOP_MAPRED_HOME=/home/ubuntu/HADOOP_FILES</value>
</property>
</configuration>
```

- for `hadoop-env.sh` for both *master* and *slaves* add `export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64`
- create the `masters` file (contains address of master nodes) for both *master* and *slaves* as

  ```
  master
  ```

- create the `workers` file (contains address of worker nodes, in hadoop 2 this file is named `slaves`) as for *master*

  ```
  master
  slave1
  slave2
  ```

  for *slaves*

  ```
  slave1
  slave2
  ```

  also remember to add the other slaves that are in `/etc/hosts`

Start the cluster

- the first time run `hadoop namenode -format`
- `HADOOP_FILES/sbin/start-all.sh`

Stop the cluster

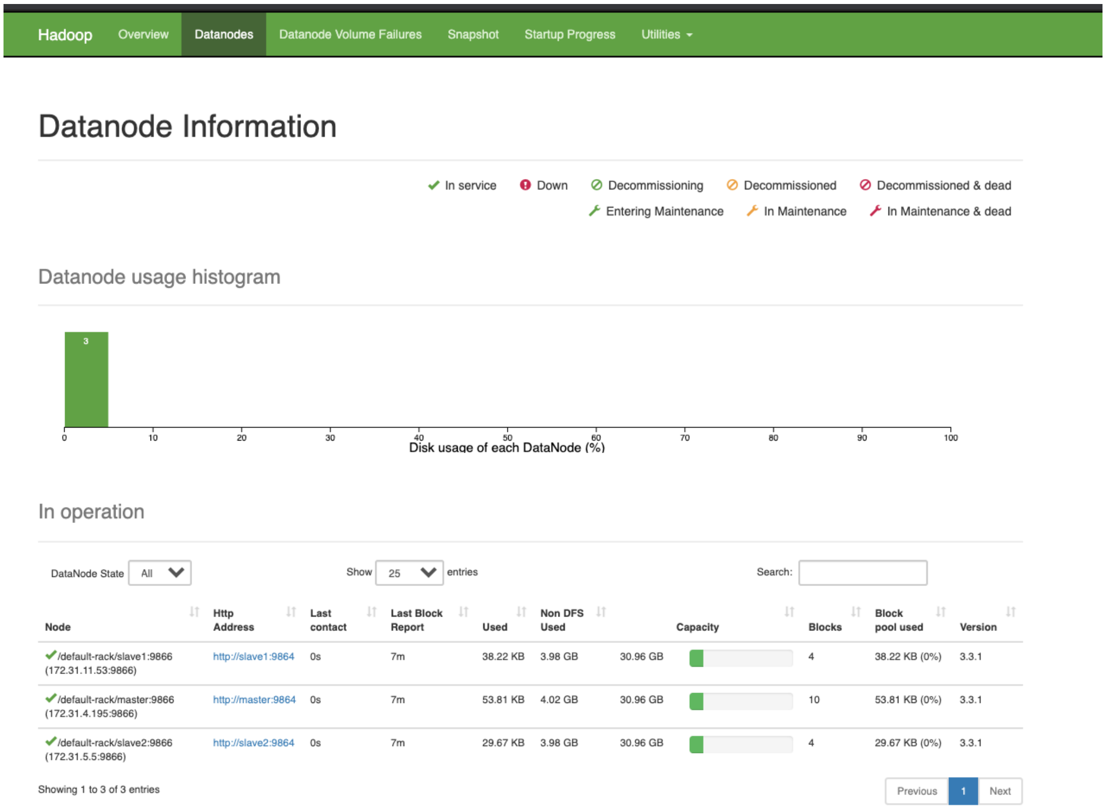
- `HADOOP_FILES/sbin/stop-all.sh`

Before restart (ONLY if there are problems)

- remember to use `jps` command in the nodes for see the active services
- `rm -r /tmp/` (consider to delete only the hadoop folders inside /tmp)
- `rm -r datanode/current` in each datanode and in the name node
- `rm -r namenode/current` in the name node

Some screens

Cluster Datanodes:



Datanode Information

✓ In service ● Down ⊘ Decommissioning ⊘ Decommissioned ⊘ Decommissioned & dead
🔧 Entering Maintenance 🔧 In Maintenance 🔧 In Maintenance & dead

Datanode usage histogram

In operation

DataNode State All

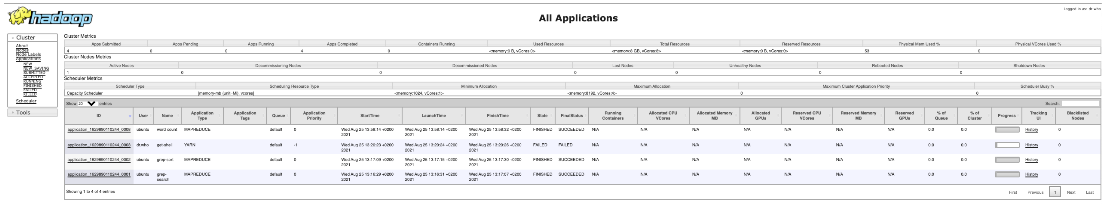
Show 25 entries

Search:

Node	Http Address	Last contact	Last Block Report	Used	Non DFS Used		Capacity	Blocks	Block pool used	Version
✓/default-rack/slave1:9866 (172.31.11.53:9866)	http://slave1:9864	0s	7m	38.22 KB	3.98 GB	30.96 GB		4	38.22 KB (0%)	3.3.1
✓/default-rack/master:9866 (172.31.4.195:9866)	http://master:9864	0s	7m	53.81 KB	4.02 GB	30.96 GB		10	53.81 KB (0%)	3.3.1
✓/default-rack/slave2:9866 (172.31.5.5:9866)	http://slave2:9864	0s	7m	29.67 KB	3.98 GB	30.96 GB		4	29.67 KB (0%)	3.3.1

Showing 1 to 3 of 3 entries

Hadoop interface (http://master:8088):



All Applications

Logged in as: dr.who

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Used Resources	Total Resources	Reserved Resources	Physical Mem Used %	Physical VCores Used %
4	0	0	4	0	<memory:0 B, vCores:0>	<memory:8 GB, vCores:8>	<memory:0 B, vCores:0>	53	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes	Shutdown Nodes
1	0	0	0	0	0	0

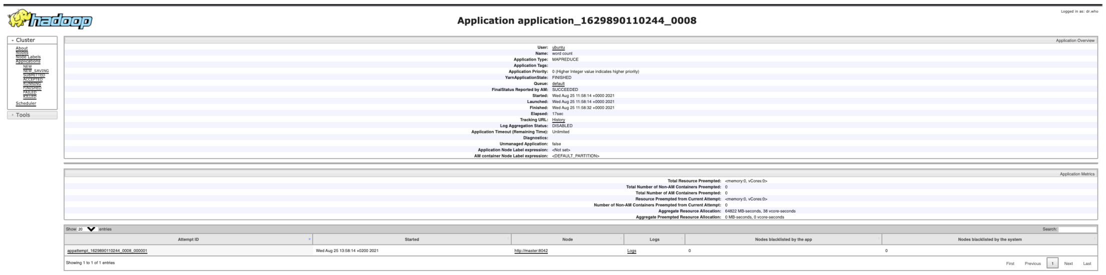
Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximum Cluster Application Priority	Scheduler Busy %
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0	0

Show 20 entries Search:

ID	User	Name	Application Type	Application Tags	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Allocated GPUs	Reserved CPU VCores	Reserved Memory MB	Reserved GPUs	% of Queue	% of Cluster	Progress	Tracking UI	Blacklisted Nodes
application_1629890110244_0008	ubuntu	word count	MAPREDUCE		default	0	Wed Aug 25 13:58:14 +0200 2021	Wed Aug 25 13:58:14 +0200 2021	Wed Aug 25 13:58:32 +0200 2021	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0		History	0
application_1629890110244_0003	dr.who	get-shell	YARN		default	-1	Wed Aug 25 13:20:23 +0200 2021	Wed Aug 25 13:20:24 +0200 2021	Wed Aug 25 13:20:26 +0200 2021	FAILED	FAILED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0		History	0
application_1629890110244_0002	ubuntu	grep-sort	MAPREDUCE		default	0	Wed Aug 25 13:17:09 +0200 2021	Wed Aug 25 13:17:15 +0200 2021	Wed Aug 25 13:17:30 +0200 2021	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0		History	0
application_1629890110244_0001	ubuntu	grep-search	MAPREDUCE		default	0	Wed Aug 25 13:16:29 +0200 2021	Wed Aug 25 13:16:31 +0200 2021	Wed Aug 25 13:17:07 +0200 2021	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0		History	0

Showing 1 to 4 of 4 entries



Application application_1629890110244_0008

Logged in as: dr.who

Application Overview

User:	ubuntu
Name:	word count
Application Type:	MAPREDUCE
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Wed Aug 25 11:58:14 +0000 2021
Launched:	Wed Aug 25 11:58:14 +0000 2021
Finished:	Wed Aug 25 11:58:32 +0000 2021
Elapsed:	17sec
Tracking URL:	History
Log Aggregation Status:	DISABLED
Application Timeout (Remaining Time):	Unlimited
Diagnostics:	
Unmanaged Application:	false
Application Node Label expression:	<Not set>
AM container Node Label expression:	<DEFAULT_PARTITION>

Application Metrics

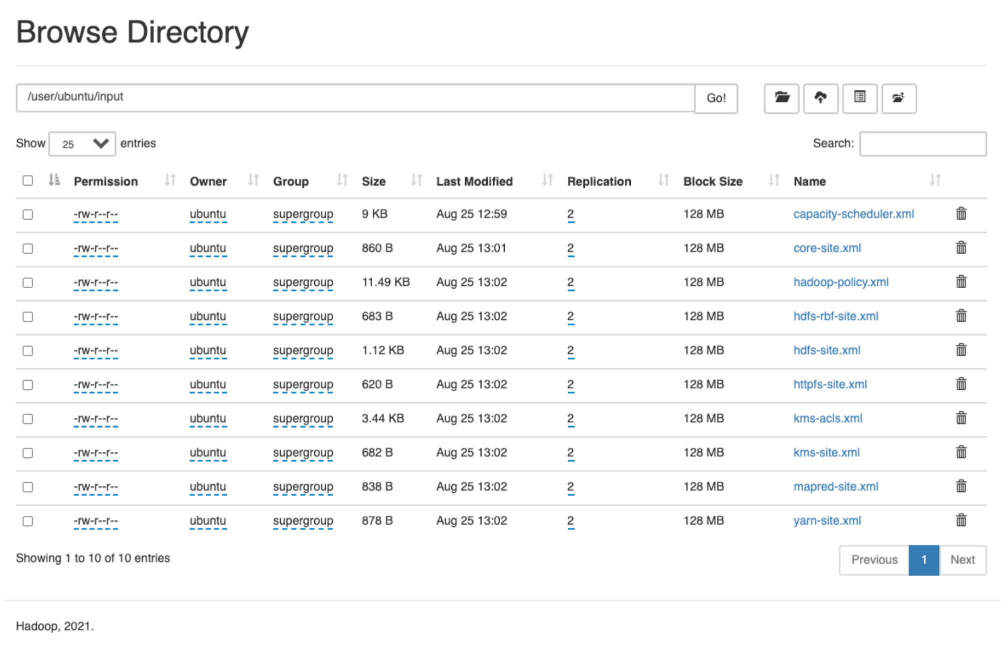
Total Resource Preempted:	<memory:0, vCores:0>
Total Number of Non-AM Containers Preempted:	0
Total Number of AM Containers Preempted:	0
Resource Preempted from Current Attempt:	<memory:0, vCores:0>
Number of Non-AM Containers Preempted from Current Attempt:	0
Aggregate Resource Allocation:	64822 MB-seconds, 38 vcore-seconds
Aggregate Preempted Resource Allocation:	0 MB-seconds, 0 vcore-seconds

Show 20 entries Search:

Attempt ID	Started	Node	Logs	Nodes blacklisted by the app	Nodes blacklisted by the system
appattempt_1629890110244_0008_000001	Wed Aug 25 13:58:14 +0200 2021	http://master:8042	Logs	0	0

Showing 1 to 1 of 1 entries

HDFS interface (http://master:9870/dfshealth.html):



Browse Directory

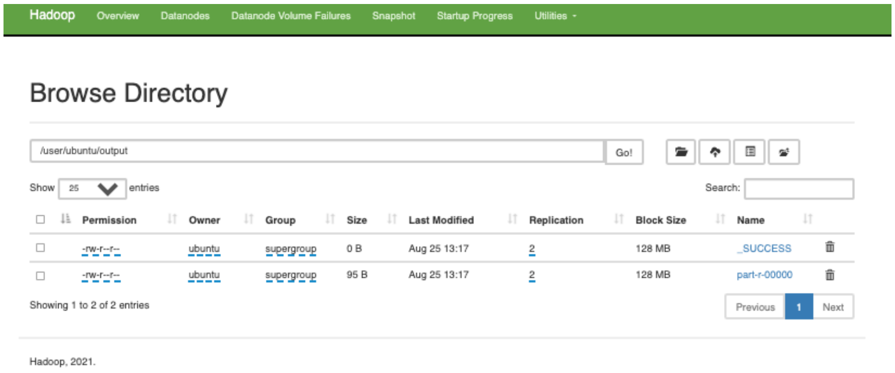
/user/ubuntu/input Go!

Show 25 entries Search:

□	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	
□	-rw-r--r--	ubuntu	supergroup	9 KB	Aug 25 12:59	2	128 MB	capacity-scheduler.xml	🗑
□	-rw-r--r--	ubuntu	supergroup	860 B	Aug 25 13:01	2	128 MB	core-site.xml	🗑
□	-rw-r--r--	ubuntu	supergroup	11.49 KB	Aug 25 13:02	2	128 MB	hadoop-policy.xml	🗑
□	-rw-r--r--	ubuntu	supergroup	683 B	Aug 25 13:02	2	128 MB	hdfs-rbf-site.xml	🗑
□	-rw-r--r--	ubuntu	supergroup	1.12 KB	Aug 25 13:02	2	128 MB	hdfs-site.xml	🗑
□	-rw-r--r--	ubuntu	supergroup	620 B	Aug 25 13:02	2	128 MB	httpfs-site.xml	🗑
□	-rw-r--r--	ubuntu	supergroup	3.44 KB	Aug 25 13:02	2	128 MB	kms-acls.xml	🗑
□	-rw-r--r--	ubuntu	supergroup	682 B	Aug 25 13:02	2	128 MB	kms-site.xml	🗑
□	-rw-r--r--	ubuntu	supergroup	838 B	Aug 25 13:02	2	128 MB	mapred-site.xml	🗑
□	-rw-r--r--	ubuntu	supergroup	878 B	Aug 25 13:02	2	128 MB	yarn-site.xml	🗑

Showing 1 to 10 of 10 entries

Hadoop, 2021.



Browse Directory

/user/ubuntu/output Go!

Show 25 entries Search:

□	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	
□	-rw-r--r--	ubuntu	supergroup	0 B	Aug 25 13:17	2	128 MB	_SUCCESS	🗑
□	-rw-r--r--	ubuntu	supergroup	95 B	Aug 25 13:17	2	128 MB	part-r-00000	🗑

Showing 1 to 2 of 2 entries

Hadoop, 2021.

Example of a job execution and HDFS interaction

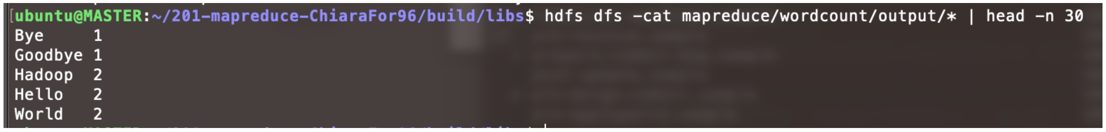
An example of a job is in `WordCount.java`, it counts for each word in a hdfs folder the number of occurrences.

We interact with hdfs with the command `hdfs dfs`, for example:

```
# Explore HDFS directories with -ls
hdfs dfs -ls /
# Create a bigdata folder in your HDFS home
hdfs dfs -mkdir bigdata
# Create a dummy file in your folder in the local file system
echo 'This is a dummy file' > dummy.txt
# Put the dummy file to your bigdata folder in HDFS
hdfs dfs -put bigdata/dummy.txt
# Get the job file from the virtual cluster's HDFS
hdfs dfs -get /path/WordCount.java
# Print hdfs file
hdfs dfs -cat mapreduce/output/* | head -n 30
```

After obtain the jar with gradle we run it with: `hadoop jar jarName.jar WordCount inputHDFSFoder outputHDFSFolder`

The result of the running:



```
ubuntu@MASTER:~/201-mapreduce-ChiaraFor96/build/libs$ hdfs dfs -cat mapreduce/wordcount/output/* | head -n 30
Bye     1
Goodbye 1
Hadoop  2
Hello   2
World   2
```

The source code of `WordCount.java`

```
package exercise1;
import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WordCount {

        public static class TokenizerMapper
        extends Mapper<Object, Text, Text, IntWritable>{

                private final static IntWritable one = new IntWritable(1);
                private Text word = new Text();

                public void map(Object key, Text value, Context context
                                ) throws IOException, InterruptedException {
                        StringTokenizer itr = new StringTokenizer(value.toString());
                        while (itr.hasMoreTokens()) {
                                word.set(itr.nextToken());
                                context.write(word, one);
                        }
                }
        }

        public static class IntSumReducer
        extends Reducer<Text,IntWritable,Text,IntWritable> {
                private IntWritable result = new IntWritable();

                public void reduce(Text key, Iterable<IntWritable> values,
                                Context context
                                ) throws IOException, InterruptedException {
                        int sum = 0;
                        for (IntWritable val : values) {
                                sum += val.get();
                        }
                        result.set(sum);
                        context.write(key, result);
                }
        }

        public static void main(String[] args) throws Exception {
                Configuration conf = new Configuration();
                Job job = Job.getInstance(conf, "word count");

                Path inputPath = new Path(args[0]), outputPath = new Path(args[1]);
                FileSystem fs = FileSystem.get(new Configuration());

                if (fs.exists(outputPath)) {
                        fs.delete(outputPath, true);
                }

                job.setJarByClass(WordCount.class);
                job.setMapperClass(TokenizerMapper.class);

                if(args.length>2){
                        if(Integer.parseInt(args[2])>=0){
                                job.setNumReduceTasks(Integer.parseInt(args[2]));
                        }
                }

                job.setReducerClass(IntSumReducer.class);
                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(IntWritable.class);

                FileInputFormat.addInputPath(job, inputPath);
                FileOutputFormat.setOutputPath(job, outputPath);

                System.exit(job.waitForCompletion(true) ? 0 : 1);
        }
}
```