SE256 :Scalable Systems for Data Science

Lab Session : 1

YARN Cluster

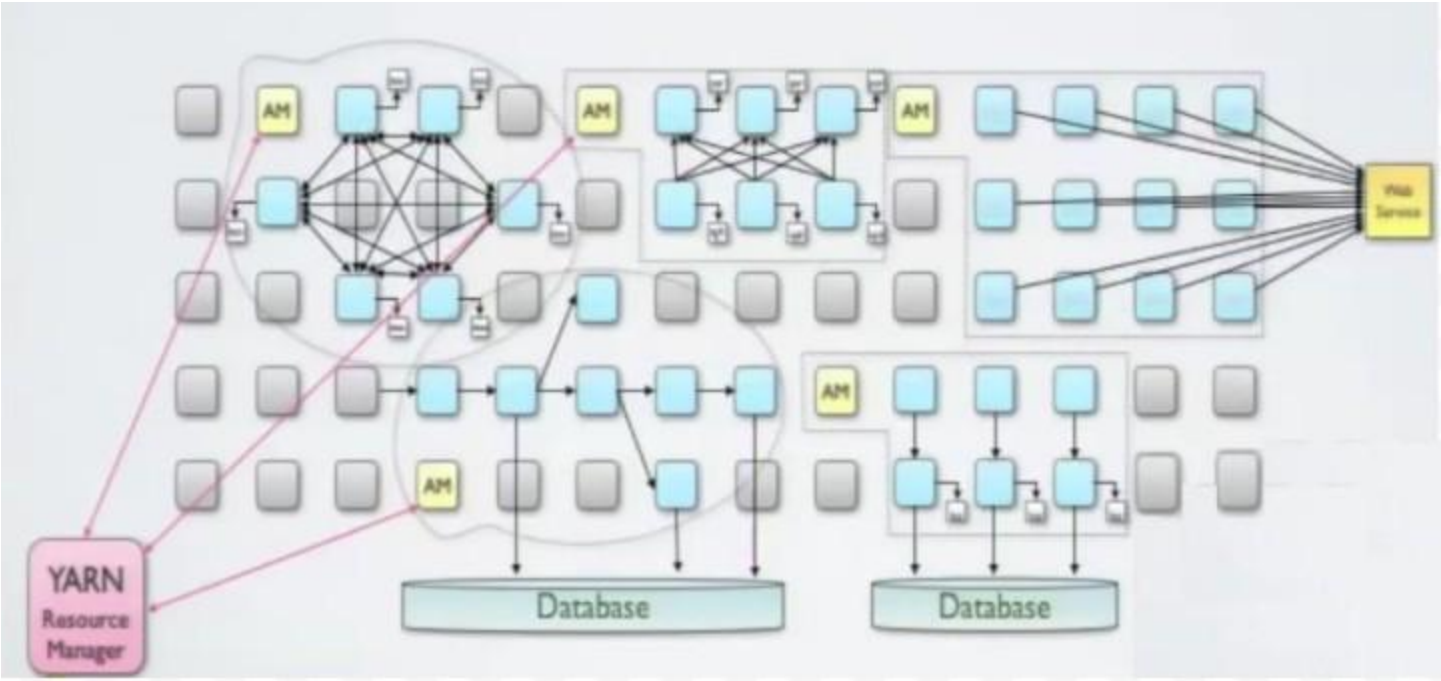


Image source : https://www.youtube.com/watch?v=HHv2pkIJjR0

YARN Config.

1. The hostname of the RM
2. Memory to allocate to each container request at the Resource Manager
3. CPU cores to allocate to each container request at the Resource Manager
4. container related settings

HDFS Config.

1. DataNode : Paths on the local filesystem where it should store its blocks
2. NameNode : Path on the local filesystem where the NameNode stores the namespace and transaction logs persistently.
3. NameNode URI

JDK Installation

Hadoop requires a working Java 5+ (Java 1.5+) installation. We now describe the installation of OpenJDK 7.

sudo apt-get install openjdk-7-jdk [Ubuntu]

su -c "yum install java-1.7.0-openjdk-devel" [Fedora]

```
dream@dream:~$ sudo apt-get install openjdk-7-jdk
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  acl at-spi2-core ca-certificates-java colord cpp cpp-4.8 dbus-x11
  dconf-gsettings-backend dconf-service desktop-file-utils fontconfig
  fontconfig-config fonts-dejavu-core fonts-dejavu-extra gconf-service
  gconf-service-backend gconf2 gconf2-common gvfs gvfs-common gvfs-daemons
  gvfs-libs hicolor-icon-theme java-common libasound2 libasound2-data
  libasyncns0 libatasmart4 libatk-bridge2.0-0 libatk-wrapper-java
  libatk-wrapper-java-jni libatk1.0-0 libatk1.0-data libatspi2.0-0
  libavahi-glib1 libbonobo2-0 libbonobo2-common libcairo-gobject2 libcairo2
  libcanberra0 libcloog-isl4 libcolord1 libcolorhug1 libdatrie1 libdconf1
  libdrm-intel1 libdrm-nouveau2 libdrm-radeon1 libexif12 libflac8
  libfontconfig1 libfontenc1 libgconf-2-4 libgconf2-4 libgd3
  libgdk-pixbuf2.0-0 libgdk-pixbuf2.0-common libgif4 libgl1-mesa-dri
  libgl1-mesa-glx libglapi-mesa libgnome2-0 libgnome2-bin libgnome2-common
  libgnomevfs2-0 libgnomevfs2-common libgphoto2-6 libgphoto2-l10n
  libgphoto2-port10 libgraphite2-3 libgtk-3-0 libgtk-3-bin libgtk-3-common
  libgtk2.0-0 libgtk2.0-bin libgtk2.0-common libgudev-1.0-0 libgusb2
  libharfbuzz0b libice-dev libice6 libidl-common libidl0 libieee1284-3
  libisl10 libjasper1 libjbig0 libjpeg-turbo8 libjpeg8 liblcms2-2 libllvm3.4
  libltdl7 libmpc3 libmpfr4 libnspr4 libnss3 libnss3-nssdb libogg0
  liborbit-2-0 liborbit2 libpango-1.0-0 libpangocairo-1.0-0 libpangoft2-1.0-0
  libpciaccess0 libpcsclite1 libpixman-1-0 libpthread-stubs0-dev libpulse0
  libsane libsane-common libsctp1 libsecret-1-0 libsecret-common libsm-dev
  libsm6 libsndfile1 libthai-data libthai0 libtiff5 libtxc-dxtn-s2tc0
```

JDK Installation

It is also necessary to add the following lines to the end of the file **$HOME/.bashrc**.

[For **64-bit** Linux]

export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64

export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$JAVA_HOME/jre/lib/amd64/server

[For **32-bit** Linux]

export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-i386

export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$JAVA_HOME/jre/lib/i386/server

Compile the file with the command **source $HOME/.bashrc**.

SSH Configuration

sudo apt-get install openssh-server **[Ubuntu]**

su -c "yum install openssh-server" **[Fedora]**

```
dream@dream:~$ sudo apt-get install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
openssh-server is already the newest version.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
dream@dream:~$
```

configure the password-less SSH connection. First, create an RSA key pair

$ ssh-keygen -t rsa

This creates files **id_rsa** and **id_rsa.pub** under the default directory **$HOME/.ssh/**. Then, we copy the public key **id_rsa.pub** to another file **authorized_keys**:

$ cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys

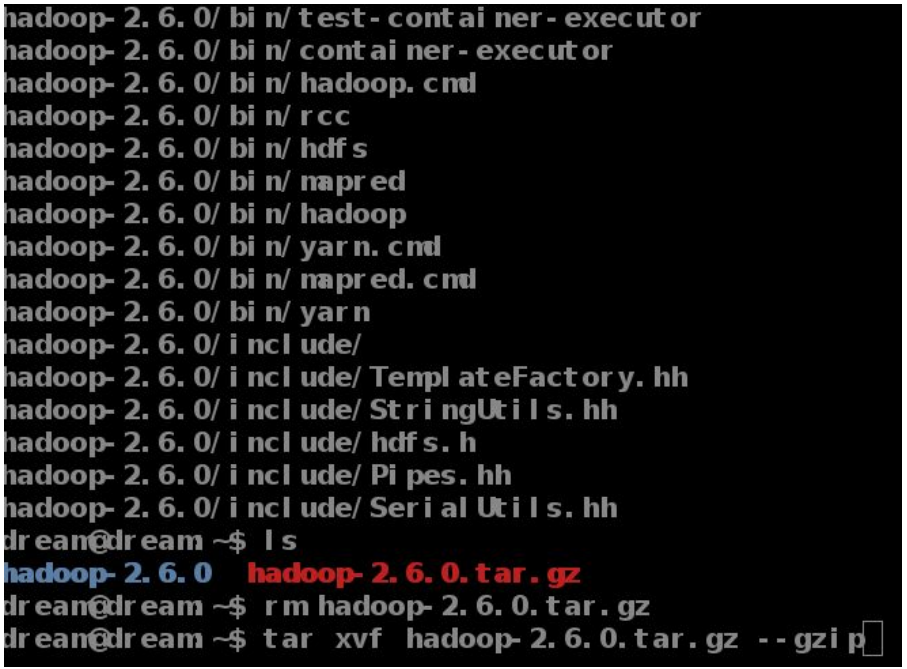
Downloading hadoop

Download hadoop using wget command:

wget http://www.eu.apache.org/dist/hadoop/common/hadoop-2.6.0/hadoop-2.6.0.tar.gz

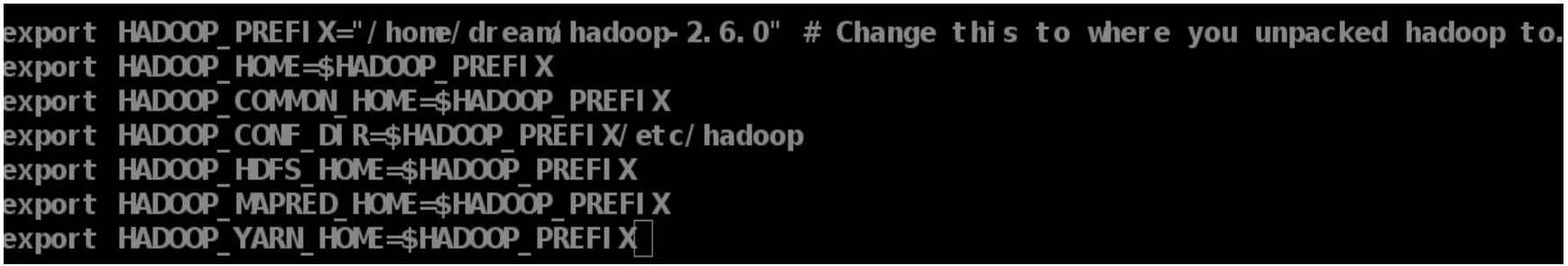
Extracting files

tar xvf hadoop-2.6.0.tar.gz --gzip



Setting hadoop env variables

```
export HADOOP_PREFIX="/home/dream/hadoop-2.6.0' # Change this to where you unpacked hadoop to.

export HADOOP_HOME=$HADOOP_PREFIX
export HADOOP_COMMON_HOME=$HADOOP_PREFIX
export HADOOP_CONF_DIR=$HADOOP_PREFIX/etc/hadoop
export HADOOP_HDFS_HOME=$HADOOP_PREFIX
export HADOOP_MAPRED_HOME=$HADOOP_PREFIX
export HADOOP_YARN_HOME=$HADOOP_PREFIX
```



Making directories required for hdfs

```
dream@dream:~$ mkdir -p hadoop-2.6.0/hdfs/datanode/ hadoop-2.6.0/hdfs/namenode
dream@dream:~$ ls hadoop-2.6.0/hdfs/
datanode  namenode
```

Configuring hdfs-site.xml

```
File  Edit  View  Search  Terminal  Help
 GNU nano 2.2.6                File: /home/dream/hadoop-2.6.0/etc/hadoop/hdfs-site.xml                Modified

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
        <name>dfs.datanode.data.dir</name>
        <value>file:///home/dream/hadoop-2.6.0/hdfs/datanode</value>
        <description>Comma separated list of paths on the local filesystem of a DataNode where it should store its blocks.</description>
    </property>

    <property>
        <name>dfs.namenode.name.dir</name>
        <value>file:///home/dream/hadoop-2.6.0/hdfs/namenode</value>
        <description>Path on the local filesystem where the NameNode stores the namespace and transaction logs persistently.</descripti$
    </property>
</configuration>
```

Configuring core-site.xml

File Edit View Search Terminal Help

GNU nano 2.2.6 File: /home/dream/hadoop-2.6.0/etc/hadoop/core-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
        <name>fs.defaultFS</name>
        <value>hdfs://localhost/</value>
        <description>NameNode URI</description>
    </property>
</configuration>
```

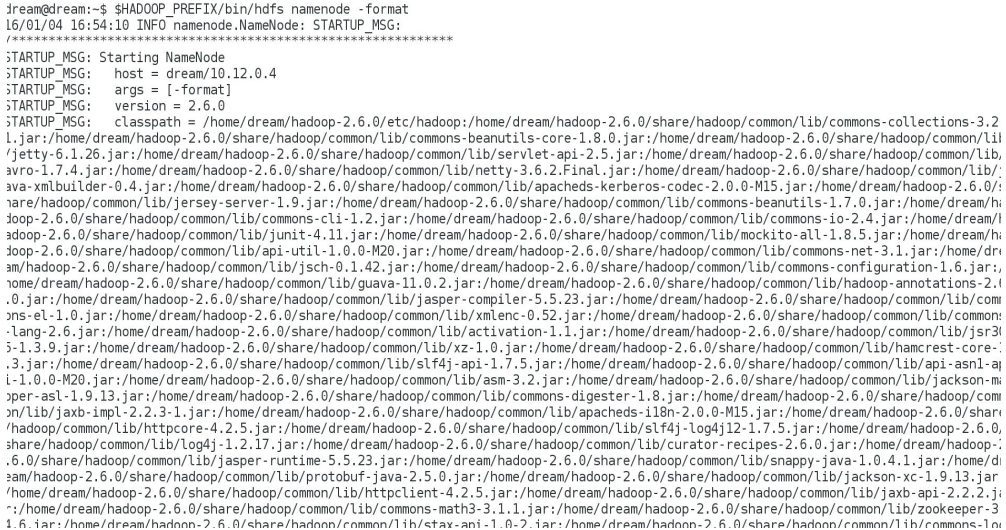
Configuring yarn-site.xml

```
 GNU nano 2.2.6                File: /home/dream/hadoop-2.6.0/etc/hadoop/yarn-site.xml                Modified

<?xml version="1.0"?>
<configuration>

<!-- Site specific YARN configuration properties -->
<property>
        <name>yarn.scheduler.minimum-allocation-mb</name>
        <value>128</value>
        <description>Minimum limit of memory to allocate to each container request at the Resource Manager.</description>
    </property>
    <property>
        <name>yarn.scheduler.maximum-allocation-mb</name>
        <value>2048</value>
        <description>Maximum limit of memory to allocate to each container request at the Resource Manager.</description>
    </property>
    <property>
        <name>yarn.scheduler.minimum-allocation-vcores</name>
        <value>1</value>
        <description>The minimum allocation for every container request at the RM, in terms of virtual CPU cores. Requests lower than t$
    </property>
    <property>
        <name>yarn.scheduler.maximum-allocation-vcores</name>
        <value>2</value>
        <description>The maximum allocation for every container request at the RM, in terms of virtual CPU cores. Requests higher than $
    </property>
    <property>
        <name>yarn.nodemanager.resource.memory-mb</name>
        <value>4096</value>
        <description>Physical memory, in MB, to be made available to running containers</description>
    </property>
    <property>
        <name>yarn.nodemanager.resource.cpu-vcores</name>
```

Starting

Now that we've finished configuring everything, it's time to setup the folders and start the daemons:

```
# Format the namenode directory (DO THIS ONLY ONCE, THE FIRST TIME)
$HADOOP_PREFIX /bin/hdfs namenode -format
# Start the namenode daemon
$HADOOP_PREFIX /sbin/hadoop-daemon.sh start namenode
# Start the datanode daemon
$HADOOP_PREFIX /sbin/hadoop-daemon.sh start datanode

## Start YARN daemons
# Start the resourcemanager daemon
$HADOOP_PREFIX /sbin/yarn-daemon.sh start resourcemanager
# Start the nodemanager daemon
$HADOOP_PREFIX /sbin/yarn-daemon.sh start nodemanager
```



Starting Hadoop daemon process

```
File  Edit  View  Search  Terminal  Help
dream@dream:~$ $HADOOP_PREFIX/sbin/hadoop-daemon.sh start namenode
starting namenode, logging to /home/dream/hadoop-2.6.0/logs/hadoop-dream-namenode-dream.out
dream@dream:~$ jps
11042 Jps
10967 NameNode
dream@dream:~$ $HADOOP_PREFIX/sbin/hadoop-daemon.sh start datanode
starting datanode, logging to /home/dream/hadoop-2.6.0/logs/hadoop-dream-datanode-dream.out
dream@dream:~$ jps
11144 Jps
10967 NameNode
11065 DataNode
dream@dream:~$ $HADOOP_PREFIX/sbin/yarn-daemon.sh start resourcemanager
starting resourcemanager, logging to /home/dream/hadoop-2.6.0/logs/yarn-dream-resourcemanager-dream.out
dream@dream:~$ jps
11403 Jps
11173 ResourceManager
10967 NameNode
11065 DataNode
dream@dream:~$ $HADOOP_PREFIX/sbin/yarn-daemon.sh start nodemanager
starting nodemanager, logging to /home/dream/hadoop-2.6.0/logs/yarn-dream-nodemanager-dream.out
dream@dream:~$ jps
11431 NodeManager
11173 ResourceManager
10967 NameNode
11525 Jps
11065 DataNode
dream@dream:~$
```

```
File  Edit  View  Search  Terminal  Help
dream@dream:~/hadoop-2.6.0$ $HADOOP_PREFIX/bin/hadoop jar ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar pi 10 100
Number of Maps  = 10
Samples per Map = 100
Wrote input for Map #0
Wrote input for Map #1
Wrote input for Map #2
Wrote input for Map #3
Wrote input for Map #4
Wrote input for Map #5
Wrote input for Map #6
Wrote input for Map #7
Wrote input for Map #8
Wrote input for Map #9
Starting Job
16/01/04 16:59:45 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.session-id
16/01/04 16:59:45 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
16/01/04 16:59:45 INFO input.FileInputFormat: Total input paths to process : 10
16/01/04 16:59:45 INFO mapreduce.JobSubmitter: number of splits:10
16/01/04 16:59:46 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local183546265_0001
16/01/04 16:59:46 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
16/01/04 16:59:46 INFO mapreduce.Job: Running job: job_local183546265_0001
16/01/04 16:59:46 INFO mapred.LocalJobRunner: OutputCommitter set in config null
16/01/04 16:59:46 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapreduce.lib.output.FileOutputCommitter
16/01/04 16:59:46 INFO mapred.LocalJobRunner: Waiting for map tasks
16/01/04 16:59:46 INFO mapred.LocalJobRunner: Starting task: attempt_local183546265_0001_m_000000_0
16/01/04 16:59:46 INFO mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
16/01/04 16:59:46 INFO mapred.MapTask: Processing split: hdfs://localhost/user/dream/QuasiMonteCarlo_1451926782358_1678543576/in/part0
+118
16/01/04 16:59:46 INFO mapred.MapTask: (EQUATOR) 0 kvi 26214396(104857584)
16/01/04 16:59:46 INFO mapred.MapTask: mapreduce.task.io.sort.mb: 100
16/01/04 16:59:46 INFO mapred.MapTask: soft limit at 83886080
16/01/04 16:59:46 INFO mapred.MapTask: bufstart = 0; bufvoid = 104857600
16/01/04 16:59:46 INFO mapred.MapTask: kvstart = 26214396; length = 6553600
16/01/04 16:59:46 INFO mapred.MapTask: Map output collector class = org.apache.hadoop.mapred.MapTask$MapOutputBuffer
16/01/04 16:59:46 INFO mapred.LocalJobRunner:
16/01/04 16:59:46 INFO mapred.MapTask: Starting flush of map output
```

```
        HDFS: Number of write operations=135
    Map-Reduce Framework
        Map input records=10
        Map output records=20
        Map output bytes=180
        Map output materialized bytes=280
        Input split bytes=1420
        Combine input records=0
        Combine output records=0
        Reduce input groups=2
        Reduce shuffle bytes=280
        Reduce input records=20
        Reduce output records=0
        Spilled Records=40
        Shuffled Maps =10
        Failed Shuffles=0
        Merged Map outputs=10
        GC time elapsed (ms)=77
        CPU time spent (ms)=0
        Physical memory (bytes) snapshot=0
        Virtual memory (bytes) snapshot=0
        Total committed heap usage (bytes)=5345640448
    Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
    File Input Format Counters
        Bytes Read=1180
    File Output Format Counters
        Bytes Written=97
Job Finished in 2.913 seconds
Estimated value of Pi is 3.14800000000000000000
```

Demo

WordCount Example

-log

-output