Steps for Hadoop installation

https://archive.apache.org/dist/hadoop/common/hadoop-2.9.1/hadoop-2.9.1.tar.gz

Download Hadoop and unzip it with Cygwin terminiinal(https://cygwin.com/install.html) using command

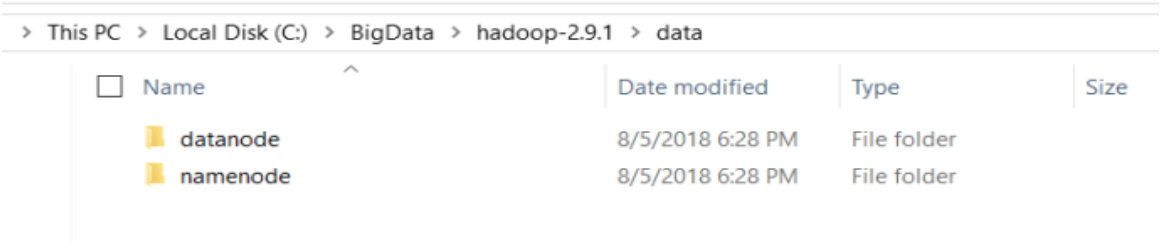
tar -xvf hadoop-2.9.1.tar.gz

Then download HadoopMaster(from teams BigData channel or GitHub) and replace the hadoop-2.9.1 bin folder with the bin folder inside HadoopMaster

Then follow the following steps for Hadoop configuration and env variable settings

Create folders for datanode and namenode

Goto C:/BigData/hadoop-2.9.1 and create a folder 'data'. Inside the 'data' folder create two folders 'datanode' and 'namenode'. Your files on HDFS will reside under the datanode folder.



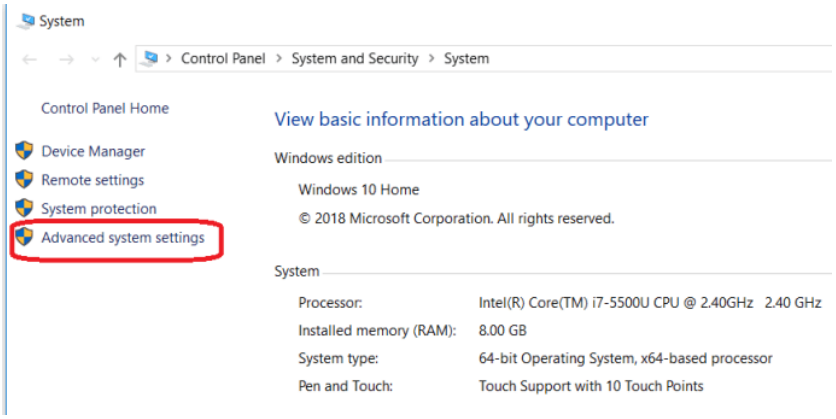
Hadoop Namenode and Datanode

Set Hadoop Environment Variables

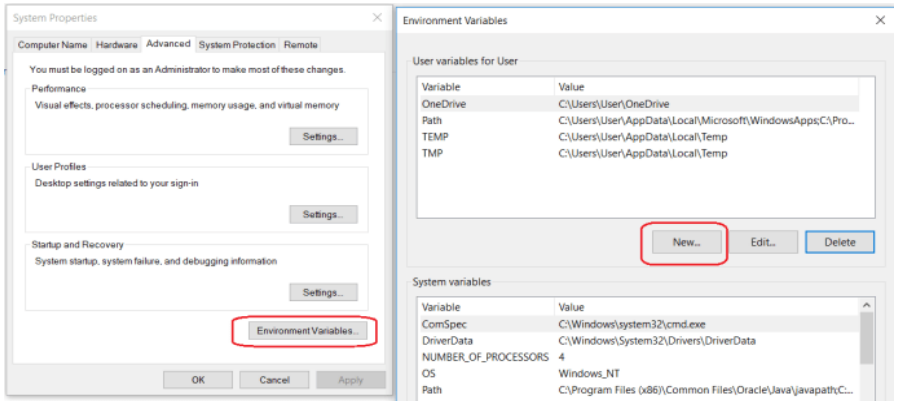
Hadoop requires following environment variables to be set.

- HADOOP_HOME="C:\BigData\hadoop-2.9.1″
- HADOOP_BIN="C:\BigData\hadoop-2.9.1\bin"
- JAVA_HOME=<Root of your JDK installation>"

To set these variables, navigate to My Computer or This PC. Right click -> Properties -> Advanced System settings -> Environment variables. Click New to create a new environment variables.

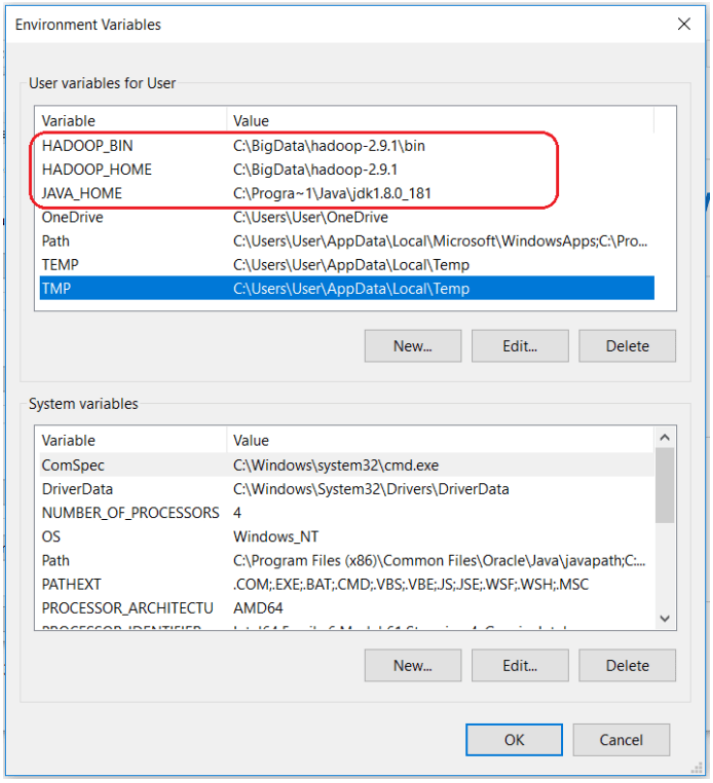


Windows Environment Variables



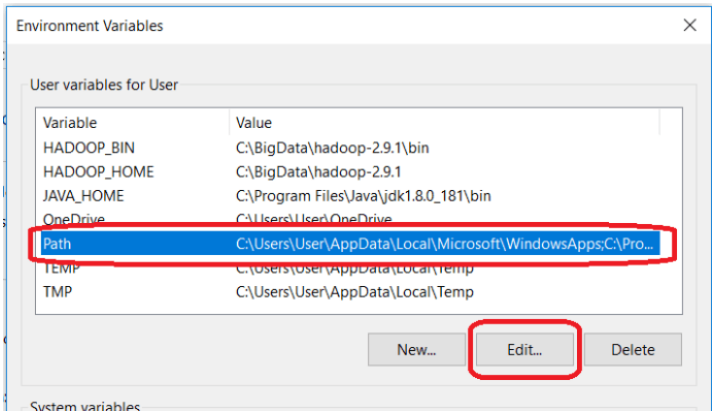
Windows Environment Variables

If you don't have JAVA 1.8 installed then you'll need to download and install it first. If JAVA_HOME environment variable is already set then check whether the path has any spaces in it (ex: *C:\Program Files\Java\…*). Spaces in the JAVA_HOME path will lead you to issues. There is a trick to get around it. Replace '*Program Files*' to '*Progra~1*' in the variable value. Ensure that the version of Java is 1.8 and JAVA_HOME is pointing to JDK 1.8.

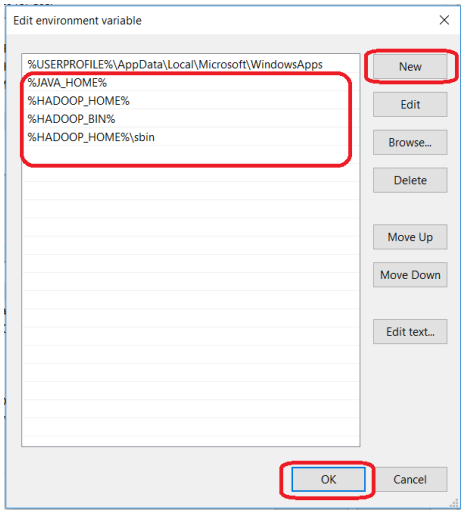


Set Hadoop Environment Variables

Edit PATH Environment Variable



Click on New and Add %JAVA_HOME%, %HADOOP_HOME%, %HADOOP_BIN%, %HADOOP_HOME%/sbin to your PATH one by one.



Set Windows PATH Variable

Now that we have set the environment variables, we need to validate them. Open a new Windows Command prompt and run echo command on each variable to confirm they are assigned the desired values.

```
echo %HADOOP_HOME%
echo %HADOOP_BIN%
echo %PATH%
```

If the variables are not initialized yet then it can probably be because you are testing them in an old session. Make sure you have opened a new command prompt to test them.

Configure Hadoop

Once environment variables are set up, we need to configure Hadoop by editing the following configurations files.

- hadoop-env.cmd
- core-site.xml
- hdfs-site.xml
- mapred-site.xml

Edit hadoop-env.cmd

First, let's configure the Hadoop environment file. Open C:\BigData\hadoop-2.9.1\etc\hadoop**hadoop-env.cmd** and add below content at the bottom

```
set HADOOP_PREFIX=%HADOOP_HOME%
set HADOOP_CONF_DIR=%HADOOP_PREFIX%\etc\hadoop
set YARN_CONF_DIR=%HADOOP_CONF_DIR%
```

```
set PATH=%PATH%;%HADOOP_PREFIX%\bin
```

Edit core-site.xml

Now, configure Hadoop Core's settings. Open C:\BigData\hadoop-2.9.1\etc\hadoop**core-site.xml** and below content within <configuration> </configuration> tags.

```
<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://0.0.0.0:19000</value>
  </property>
</configuration>
```

Edit hdfs-site.xml

After editing core-site.xml, you need to set replication factor and the location of namenode and datanodes. Open C:\BigData\hadoop-2.9.1\etc\hadoop**hdfs-site.xml** and below content within <configuration> </configuration> tags.

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>C:\BigData\hadoop-2.9.1\data\namenode</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>C:\BigData\hadoop-2.9.1\data\datanode</value>
  </property>
</configuration>
```

Edit mapred-site.xml

Finally, let's configure properties for the Map-Reduce framework. Open C:\BigData\hadoop-2.9.1\etc\hadoop**mapred-site.xml** and below content within <configuration> </configuration> tags. If you don't see mapred-site.xml then open mapred-site.xml.template file and rename it to mapred-site.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<configuration>
```

```
  <property>
   <name>mapred.job.tracker</name>
   <value>master1:8021</value>
  </property>
  <property>
   <name>mapred.local.dir</name>

<value>/home/vagrant/hadoop_home/data/1/mapred/local,/home/vagrant/hadoop_home/data/
2/mapred/local,/home/vagrant/hadoop_home/data/3/mapred/local</value>

  </property>
  <property>
   <name>mapreduce.jobtracker.restart.recover</name>
   <value>true</value>
  </property>
  <!-- The web UI will bind -->
  <property>
      <name>mapred.job.tracker.http.address</name>
      <value></value>
  </property>

  <!-- This identifies the mesos-master. E.g.
zk://1.1.1.1:2181,2.2.2.2:2181,3.3.3.3:2181/mesos -->
  <property>
      <name>mapred.mesos.master</name>
      <value>zk://master1:2181,slave1:2181,slave2:2181/mesos</value>
  </property>
    <!--
      This property identifies the location of the modified hadoop distribution containing this
XML file.
      The mesos slave will download this distribution if a hadoop job is launched, extract the
file and use the hadoop binary
      to start the task tracker.
      Sample hdfs://<hdfs-namenode-host & optional port>/hadoop-2.0.0-mr1-cdh4.2.1.tgz ->
hdfs://namenode.mesosphere.io:9000/hadoop-2.0.0-mr1-cdh4.2.1.tgz
    -->
    <property>
        <name>mapred.mesos.executor.uri</name>
        <value>hdfs://master1/hadoop-2.0.0-mr1-cdh4.2.1.tgz</value>
    </property>
    <!--
      The remaining properties do not require adjustment, but for running production jobs it's
recommended to modify them
      to optimize for different cluster & machine sizes.
    -->
    <property>
```

```
    <name>mapred.mesos.slot.cpus</name>
    <value>0.20</value>
  </property>
  <property>
    <name>mapred.mesos.slot.disk</name>
    <!-- The value is in MB. -->
    <value>512</value>
  </property>
  <property>
    <name>mapred.mesos.slot.mem</name>
    <!-- Note that this is the total memory required for
     JVM overhead (256 MB) and the heap (-Xmx) of the task.
     The value is in MB. -->
    <value>368</value>
  </property>
  <property>
    <name>mapred.mesos.tasktracker.mem</name>
    <value>368</value>
  </property>
  <property>
    <name>mapred.mesos.total.map.slots.minimum</name>
    <value>1</value>
  </property>
  <property>
    <name>mapred.mesos.total.reduce.slots.minimum</name>
    <value>1</value>
  </property>
  <!-- The values below should work out of the box but you might want to optimize some of
them for running production jobs -->
  <property>
    <name>mapred.jobtracker.taskScheduler</name>
    <value>org.apache.hadoop.mapred.MesosScheduler</value>
  </property>
  <property>
    <name>mapred.mesos.taskScheduler</name>
    <value>org.apache.hadoop.mapred.JobQueueTaskScheduler</value>
  </property>
  <!-- The MesosScheduler will record some stats in this file -->
  <property>
    <name>mapred.mesos.state.file</name>
    <value>/tmp/jobtracker-state</value>
  </property>
  <!-- This is only relevant if a fixed slot policy is used -->
  <property>
    <name>mapred.tasktracker.map.tasks.maximum</name>
    <value>10</value>
```

```
</property>
<!-- This is only relevant if a fixed slot policy is used -->
<property>
   <name>mapred.tasktracker.reduce.tasks.maximum</name>
   <value>10</value>
</property>
<property>
   <name>mapreduce.jobtracker.expire.trackers.interval</name>
   <value>60000</value>
</property>
<property>
   <name>mapred.tasktracker.expiry.interval</name>
   <value>60000</value>
</property>
<property>
   <name>mapreduce.jobtracker.restart.recover</name>
   <value>true</value>
</property>
<property>
   <name>mapred.child.java.opts</name>
   <value>-XX:+UseParallelGC -Xmx256m</value>
</property>
<property>
   <name>mapreduce.tasktracker.dns.interface</name>
   <value>eth0</value>
</property>
<!-- The reduce tasks start when 60% of the maps are done -->
<property>
   <name>mapreduce.job.reduce.slowstart.completedmaps</name>
   <value>0.60</value>
</property>
<property>
   <name>mapred.reduce.slowstart.completed.maps</name>
   <value>0.60</value>
</property>
<!-- This is important when the tasktracker serves tons of maps, TODO(*) templetize -->
<property>
   <name>mapreduce.tasktracker.http.threads</name>
   <value>8</value>
</property>

<property>
   <name>tasktracker.http.threads</name>
   <value>8</value>
</property>
<property>
```

```
    <name>mapreduce.reduce.shuffle.parallelcopies</name>
    <value>20</value>
</property>
<property>
    <name>mapred.reduce.parallel.copies</name>
    <value>20</value>
</property>
<property>
    <name>mapreduce.jobtracker.handler.count</name>
    <value>70</value>
</property>
<property>
    <name>mapred.job.tracker.handler.count</name>
    <value>70</value>
</property>
<property>
    <name>mapreduce.reduce.shuffle.retry-delay.max.ms</name>
    <value>10000</value>
</property>
<property>
    <name>mapreduce.reduce.shuffle.connect.timeout</name>
    <value>10000</value>
</property>
<property>
    <name>mapreduce.reduce.shuffle.read.timeout</name>
    <value>10000</value>
</property>
<property>
    <name>mapreduce.reduce.shuffle.maxfetchfailures</name>
    <value>4</value>
</property>
<property>
    <name>mapreduce.reduce.shuffle.notify.readerror</name>
    <value>true</value>
</property>
<property>
    <name>mapreduce.map.output.compress</name>
    <value>true</value>
</property>
<property>
    <name>mapreduce.task.io.sort.mb</name>
    <value>30</value>
</property>
<property>
    <name>io.sort.mb</name>
    <value>30</value>
```

```
</property>
<property>
   <name>mapreduce.task.io.sort.factor</name>
   <value>10</value>
</property>
<property>
   <name>io.sort.factor</name>
   <value>10</value>
</property>
<property>
   <name>mapreduce.job.jvm.numtasks</name>
   <value>-1</value>
</property>
<property>
   <name>mapred.job.reuse.jvm.num.tasks</name>
   <value>-1</value>
</property>
<property>
   <name>mapreduce.job.ubertask.enable</name>
   <value>true</value>
</property>
<property>
   <name>mapreduce.job.speculative.speculativecap</name>
   <value>0.01</value>
</property>
<property>
   <name>webinterface.private.actions</name>
   <value>true</value>
</property>
<property>
   <name>mapreduce.jobtracker.webinterface.trusted</name>
   <value>true</value>
</property>
<property>
   <name>mapred.reduce.max.attempts</name>
   <value>6</value>
</property>
<property>
   <name>mapred.map.max.attempts</name>
   <value>6</value>
</property>

<property>
   <name>mapreduce.map.maxattempts</name>
   <value>6</value>
</property>
```

```
  <property>
    <name>mapreduce.reduce.maxattempts</name>
    <value>6</value>
  </property>
  <property>
    <name>mapred.max.tracker.failures</name>
    <value>6</value>
  </property>
  <property>
    <name>mapreduce.job.maxtaskfailures.per.tracker</name>
    <value>6</value>
  </property>
  <property>
    <name>mapreduce.reduce.merge.memtomem.enabled</name>
    <value>true</value>
  </property>
  <property>
    <name>mapred.skip.map.max.skip.records</name>
    <value>10</value>
  </property>
  <property>
    <name>mapreduce.map.skip.maxrecords</name>
    <value>10</value>
  </property>
  <property>
    <name>mapreduce.reduce.skip.maxgroups</name>
    <value>2</value>
  </property>
  <property>
    <name>mapred.skip.reduce.max.skip.groups</name>
    <value>2</value>
  </property>
  <property>
    <name>mapreduce.fileoutputcommitter.marksuccessfuljobs</name>
    <value>false</value>
  </property>
  <property>
    <name>mapred.mesos.tasktracker.cpus</name>
    <!-- This is the number of CPUs reserved for the container.-->
    <value>0.15</value>
  </property>
</configuration>
```

Now follow the commands to confirm Hadoop installation completion

Run CMD prompt as administrator

```
C:\Windows\system32>start-all.cmd
```

This Command will open 4 cmd prompt as hadoop datanode, hadoop namenode, yarn resourcemanager and yarn nodemanager minimize them and continue with the commands below

```
C:\Windows\system32>hdfs dfs -mkdir /test
C:\Windows\system32>hdfs dfs -ls /
```

If the output list of above command contains **test file** the Hadoop is installed successfully