Hadoop: Setting up a Single Node Cluster.

Purpose

This document describes how to set up and configure a single-node Hadoop installation so that you can quickly perform simple operations using Hadoop MapReduce and the Hadoop Distributed File System (HDFS).

Prerequisites

Supported Platforms

- GNU/Linux is supported as a development and production platform. Hadoop has been demonstrated on GNU/Linux clusters with 2000 nodes.
- Windows is also a supported platform but the followings steps are for Linux only. To set up Hadoop on Windows, see wiki page .

Required Software

Required software for Linux include:

1. Java™ must be installed. Recommended Java versions are described at HadoopJavaVersions .
2. ssh must be installed and sshd must be running to use the Hadoop scripts that manage remote Hadoop daemons if the optional start and stop scripts are to be used. Additionally, it is recommmended that pdsh also be installed for better ssh resource management.

Installing Software

If your cluster doesn't have the requisite software you will need to install it.

For example on Ubuntu Linux:

```
$ sudo apt-get install ssh
$ sudo apt-get install pdsh
```

Download

To get a Hadoop distribution, download a recent stable release from one of the Apache Download Mirrors .

Prepare to Start the Hadoop Cluster

Unpack the downloaded Hadoop distribution. In the distribution, edit the file `etc/hadoop/hadoop-env.sh` to define some parameters as follows:

```
# set to the root of your Java installation
export JAVA_HOME=/usr/java/latest
```

Try the following command:

```
$ bin/hadoop
```

This will display the usage documentation for the hadoop script.

Now you are ready to start your Hadoop cluster in one of the three supported modes:

- Local (Standalone) Mode
- Pseudo-Distributed Mode
- Fully-Distributed Mode

Standalone Operation

By default, Hadoop is configured to run in a non-distributed mode, as a single Java process. This is useful for debugging.

The following example copies the unpacked conf directory to use as input and then finds and displays every match of the given regular expression. Output is written to the given output directory.

```
$ mkdir input
$ cp etc/hadoop/*.xml input
$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-3.0.0-alpha2.ja
$ cat output/*
```

Pseudo-Distributed Operation

Hadoop can also be run on a single-node in a pseudo-distributed mode where each Hadoop daemon runs in a separate Java process.

Configuration

Use the following:

etc/hadoop/core-site.xml:

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

etc/hadoop/hdfs-site.xml:

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

Setup passphraseless ssh

Now check that you can ssh to the localhost without a passphrase:

```
$ ssh localhost
```

If you cannot ssh to localhost without a passphrase, execute the following commands:

```
$ ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
$ chmod 0600 ~/.ssh/authorized_keys
```

Execution

The following instructions are to run a MapReduce job locally. If you want to execute a job on YARN, see YARN on Single Node.

1. Format the filesystem:

```
$ bin/hdfs namenode -format
```

2. Start NameNode daemon and DataNode daemon:

```
$ sbin/start-dfs.sh
```

The hadoop daemon log output is written to the `$HADOOP_LOG_DIR` directory (defaults to `$HADOOP_HOME/logs`).

3. Browse the web interface for the NameNode; by default it is available at:
    - NameNode - `http://localhost:9870/`
4. Make the HDFS directories required to execute MapReduce jobs:

```
$ bin/hdfs dfs -mkdir /user
$ bin/hdfs dfs -mkdir /user/<username>
```

5. Copy the input files into the distributed filesystem:

```
$ bin/hdfs dfs -mkdir input
$ bin/hdfs dfs -put etc/hadoop/*.xml input
```

6. Run some of the examples provided:

```
$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-3.0.0-alpha2.ja
```

7. Examine the output files: Copy the output files from the distributed filesystem to the local filesystem and examine them:

```
$ bin/hdfs dfs -get output output
$ cat output/*
```

or

View the output files on the distributed filesystem:

```
$ bin/hdfs dfs -cat output/*
```

8. When you’re done, stop the daemons with:

```
$ sbin/stop-dfs.sh
```

YARN on a Single Node

You can run a MapReduce job on YARN in a pseudo-distributed mode by setting a few parameters and running ResourceManager daemon and NodeManager daemon in addition.

The following instructions assume that 1. ~ 4. steps of the above instructions are already executed.

1. Configure parameters as follows:

   `etc/hadoop/mapred-site.xml`:

```
<configuration>
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
</configuration>
```

   `etc/hadoop/yarn-site.xml`:

```
<configuration>
    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>
    <property>
        <name>yarn.nodemanager.env-whitelist</name>
        <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSP
    </property>
</configuration>
```

2. Start ResourceManager daemon and NodeManager daemon:

```
  $ sbin/start-yarn.sh
```

3. Browse the web interface for the ResourceManager; by default it is available at:

   - ResourceManager - `http://localhost:8088/`

4. Run a MapReduce job.

5. When you’re done, stop the daemons with:

```
  $ sbin/stop-yarn.sh
```

Fully-Distributed Operation

For information on setting up fully-distributed, non-trivial clusters see Cluster Setup.