Michael G. Noll

Applied Research. Big Data. Distributed Systems. Open Source.

- RSS

Search

Navigate…

- Blog
- Archive
- Tutorials
- Projects
- Publications

Running Hadoop on Ubuntu Linux (Multi-Node Cluster)

Table of Contents

In this tutorial I will describe the required steps for setting up a *distributed, multi-node* Apache Hadoop cluster backed by the Hadoop Distributed File System (HDFS), running on Ubuntu Linux.

Are you looking for the single-node cluster tutorial? Just head over there.

Hadoop is a framework written in Java for running applications on large clusters of commodity hardware and incorporates features similar to those of the Google File System (GFS) and of the MapReduce computing paradigm. Hadoop’s HDFS is a highly fault-tolerant distributed file system and, like Hadoop in general, designed to be deployed on low-cost hardware. It provides high throughput access to

In a previous tutorial, I described how to setup up a Hadoop single-node cluster on an Ubuntu box. The main goal of *this* tutorial is to get a more sophisticated Hadoop installation up and running, namely building a multi-node cluster using two Ubuntu boxes.

This tutorial has been tested with the following software versions:

- Ubuntu Linux 10.04 LTS (deprecated: 8.10 LTS, 8.04, 7.10, 7.04)
- Hadoop 1.0.3, released May 2012

Figure 1: Cluster of machines running Hadoop at Yahoo! (Source: Yahoo!)

Tutorial approach and structure

From two single-node clusters to a multi-node cluster – We will build a multi-node cluster using two Ubuntu boxes in this tutorial. In my humble opinion, the best way to do this for starters is to install, configure and test a "local" Hadoop setup for each of the two Ubuntu boxes, and in a second step to "merge" these two single-node clusters into one multi-node cluster in which one Ubuntu box will become the designated master (but also act as a slave with regard to data storage and processing), and the other box will become only a slave. It's much easier to track down any problems you might encounter due to the reduced complexity of doing a single-node cluster setup first on each machine.

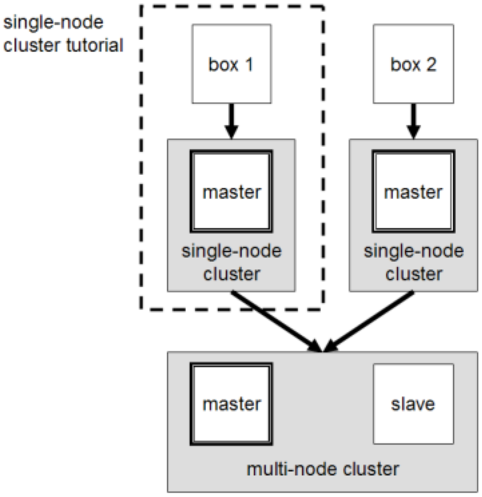

Figure 2: Tutorial approach and structure

Let's get started!

Prerequisites

Configuring single-node clusters first

The tutorial approach outlined above means that you should read now my previous tutorial on how to setup up a Hadoop single-node cluster and follow the steps described there to build a single-node Hadoop cluster on each of the two Ubuntu boxes. It is recommended that you use the ''same settings'' (e.g., installation locations and paths) on both machines, or otherwise you might run into problems later when we will migrate the two machines to the final multi-node cluster setup.

Just keep in mind when setting up the single-node clusters that we will later connect and "merge" the two machines, so pick reasonable network settings etc. now for a smooth transition later.

Done? Let's continue then!

Now that you have two single-node clusters up and running, we will modify the Hadoop configuration to make one Ubuntu box the "master" (which will also act as a slave) and the other Ubuntu box a "slave".

Note: We will call the designated master machine just the "master" from now on and the slave-only machine the "slave". We will also give the two machines these respective hostnames in their networking setup, most notably in "/etc/hosts". If the hostnames of your machines are different (e.g. "node01") then you must adapt the settings in this tutorial as appropriate.

Shutdown each single-node cluster with `bin/stop-all.sh` before continuing if you haven't done so already.

Networking

This should come hardly as a surprise, but for the sake of completeness I have to point out that both machines must be able to reach each other over the network. The easiest is to put both machines in the same network with regard to hardware and software configuration, for example connect both machines via a single hub or switch and configure the network interfaces to use a common network such as `192.168.0.x/24`.

To make it simple, we will assign the IP address `192.168.0.1` to the `master` machine and `192.168.0.2` to the `slave` machine. Update `/etc/hosts` on both machines with the following lines:

/etc/hosts (for master AND slave)

```
192.168.0.1    master
192.168.0.2    slave
```

SSH access

The `hduser` user on the `master` (aka `hduser@master`) must be able to connect a) to its own user account on the `master` – i.e. `ssh master` in this context and not necessarily `ssh localhost` – and b) to the `hduser` user account on the `slave` (aka `hduser@slave`) via a password-less SSH login. If you followed my single-node cluster tutorial, you just have to add the `hduser@master`'s public SSH key (which should be in `$HOME/.ssh/id_rsa.pub`) to the `authorized_keys` file of `hduser@slave` (in this user's `$HOME/.ssh/authorized_keys`). You can do this manually or use the following SSH command:

Distribute the SSH public key of hduser@master

```
hduser@master:~$ ssh-copy-id -i $HOME/.ssh/id_rsa.pub hduser@slave
```

This command will prompt you for the login password for user `hduser` on `slave`, then copy the public SSH key for you, creating the correct directory and fixing the permissions as necessary.

The final step is to test the SSH setup by connecting with user `hduser` from the `master` to the user account `hduser` on the `slave`. The step is also needed to save `slave`'s host key fingerprint to the `hduser@master`'s `known_hosts` file.

So, connecting from `master` to `master`…

```
hduser@master:~$ ssh master
The authenticity of host 'master (192.168.0.1)' can't be established.
RSA key fingerprint is 3b:21:b3:c0:21:5c:7c:54:2f:1e:2d:96:79:eb:7f:95.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'master' (RSA) to the list of known hosts.
Linux master 2.6.20-16-386 #2 Thu Jun 7 20:16:13 UTC 2007 i686
...
hduser@master:~$
```

…and from `master` to `slave`.

```
hduser@master:~$ ssh slave
The authenticity of host 'slave (192.168.0.2)' can't be established.
RSA key fingerprint is 74:d7:61:86:db:86:8f:31:90:9c:68:b0:13:88:52:72.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'slave' (RSA) to the list of known hosts.
Ubuntu 10.04
...
hduser@slave:~$
```

Hadoop

Cluster Overview (aka the goal)

The next sections will describe how to configure one Ubuntu box as a master node and the other Ubuntu box as a slave node. The master node will also act as a slave because we only have two machines available in our cluster but still want to spread data storage and processing to multiple machines.

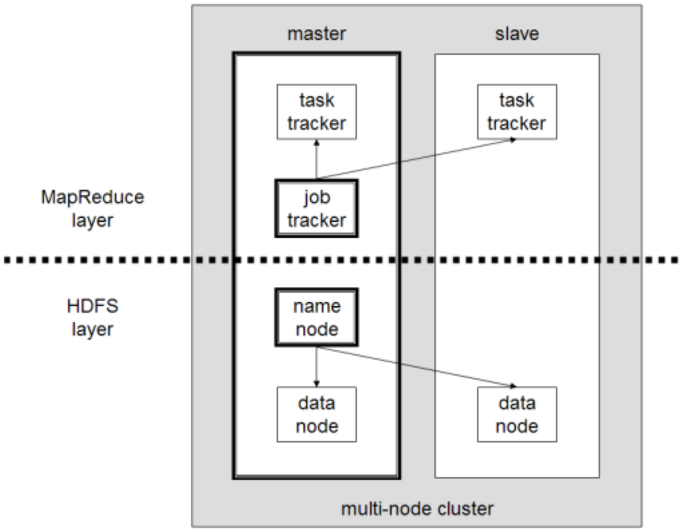

Figure 3: How the final multi-node cluster will look like

The master node will run the "master" daemons for each layer: NameNode for the HDFS storage layer, and JobTracker for the MapReduce processing layer. Both machines will run the "slave" daemons: DataNode for the HDFS layer, and TaskTracker for MapReduce processing layer. Basically, the "master" daemons are responsible for coordination and management of the "slave" daemons while the latter will do the actual data storage and data processing work.

Masters vs. Slaves

> Typically one machine in the cluster is designated as the NameNode and another machine the as JobTracker, exclusively. These are the actual "master nodes". The rest of the machines in the cluster act as both DataNode and TaskTracker. These are the slaves or "worker nodes".
>
> **Hadoop 1.x documentation** hadoop.apache.org/common/docs/...

Configuration

conf/masters (`master` only)

Despite its name, the `conf/masters` file defines on which machines Hadoop will start *secondary NameNodes* in our multi-node cluster. In our case, this is just the `master` machine. The primary NameNode and the JobTracker will always be the machines on which you run the `bin/start-dfs.sh` and `bin/start-mapred.sh` scripts, respectively (the primary NameNode and the JobTracker will be started on the same machine if you run `bin/start-all.sh`).

Note: You can also start an Hadoop daemon manually on a machine via `bin/hadoop-daemon.sh start [namenode | secondarynamenode | datanode | jobtracker | tasktracker]`, which will not take the "conf/masters" and "conf/slaves" files into account.

Here are more details regarding the `conf/masters` file:

> The secondary NameNode merges the fsimage and the edits log files periodically and keeps edits log size within a limit. It is usually run on a different machine than the primary NameNode since its memory requirements are on the same order as the primary NameNode. The secondary NameNode is started by "bin/start-dfs.sh" on the nodes specified in "conf/masters" file.
>
> **Hadoop HDFS user guide** hadoop.apache.org/common/docs/...

Again, the machine on which `bin/start-dfs.sh` is run will become the *primary* NameNode.

On `master`, update `conf/masters` that it looks like this:

conf/masters (on master)

```
master
```

conf/slaves (`master` only)

The `conf/slaves` file lists the hosts, one per line, where the Hadoop slave daemons (DataNodes and TaskTrackers) will be run. We want both the `master` box and the `slave` box to act as Hadoop slaves because we want both of them to store and process data.

On `master`, update `conf/slaves` that it looks like this:

conf/slaves (on master)

```
master
slave
```

If you have additional slave nodes, just add them to the `conf/slaves` file, one hostname per line.

conf/slaves (on master)

```
master
slave
anotherslave01
anotherslave02
anotherslave03
```

Note: The `conf/slaves` file on `master` is used only by the scripts like `bin/start-dfs.sh` or `bin/stop-dfs.sh`. For example, if you want to add DataNodes on the fly (which is not described in this tutorial yet), you can “manually” start the DataNode daemon on a new slave machine via `bin/hadoop-daemon.sh start datanode`. Using the `conf/slaves` file on the master simply helps you to make “full” cluster restarts easier.

conf/*-site.xml (all machines)

You must change the configuration files `conf/core-site.xml`, `conf/mapred-site.xml` and `conf/hdfs-site.xml` on ALL machines as follows.

First, we have to change the fs.default.name parameter (in `conf/core-site.xml`), which specifies the NameNode (the HDFS master) host and port. In our case, this is the `master` machine.

conf/core-site.xml (ALL machines)

```
<property>
  <name>fs.default.name</name>
  <value>hdfs://master:54310</value>
  <description>The name of the default file system.  A URI whose
  scheme and authority determine the FileSystem implementation.  The
  uri's scheme determines the config property (fs.SCHEME.impl) naming
  the FileSystem implementation class.  The uri's authority is used to
  determine the host, port, etc. for a filesystem.</description>
</property>
```

Second, we have to change the mapred.job.tracker parameter (in `conf/mapred-site.xml`), which specifies the JobTracker (MapReduce master) host and port. Again, this is the `master` in our case.

conf/mapred-site.xml (ALL machines)

```
<property>
  <name>mapred.job.tracker</name>
  <value>master:54311</value>
  <description>The host and port that the MapReduce job tracker runs
  at.  If "local", then jobs are run in-process as a single map
  and reduce task.
  </description>
</property>
```

Third, we change the dfs.replication parameter (in `conf/hdfs-site.xml`) which specifies the default block replication. It defines how many machines a single file should be replicated to before it becomes available. If you set this to a value higher than the number of available slave nodes (more precisely, the number of DataNodes), you will start seeing a lot of “(Zero targets found, forbidden1.size=1)” type errors in the log files.

The default value of `dfs.replication` is `3`. However, we have only two nodes available, so we set `dfs.replication` to `2`.

conf/hdfs-site.xml (ALL machines)

```
<property>
  <name>dfs.replication</name>
  <value>2</value>
  <description>Default block replication.
  The actual number of replications can be specified when the file is created.
  The default is used if replication is not specified in create time.
  </description>
</property>
```

Additional Settings

There are some other configuration options worth studying. The following information is taken from the Hadoop API Overview.

In file `conf/mapred-site.xml`:

“mapred.local.dir“

Determines where temporary MapReduce data is written. It also may be a list of directories.

“mapred.map.tasks“

As a rule of thumb, use 10x the number of slaves (i.e., number of TaskTrackers).

“mapred.reduce.tasks“

As a rule of thumb, use num_tasktrackers * num_reduce_slots_per_tasktracker * 0.99. If num_tasktrackers is small (as in the case of this tutorial), use (num_tasktrackers - 1) * num_reduce_slots_per_tasktracker.

Formatting the HDFS filesystem via the NameNode

Before we start our new multi-node cluster, we must format Hadoop's distributed filesystem (HDFS) via the NameNode. You need to do this the first time you set up an Hadoop cluster.

Warning: Do not format a running cluster because this will erase all existing data in the HDFS filesytem!

To format the filesystem (which simply initializes the directory specified by the `dfs.name.dir` variable on the NameNode), run the command

Format the cluster's HDFS file system

```
hduser@master:/usr/local/hadoop$ bin/hadoop namenode -format
... INFO dfs.Storage: Storage directory /app/hadoop/tmp/dfs/name has been successfully formatted.
hduser@master:/usr/local/hadoop$
```

Background: The HDFS name table is stored on the NameNode's (here: `master`) local filesystem in the directory specified by `dfs.name.dir`. The name table is used by the NameNode to store tracking and coordination information for the DataNodes.

Starting the multi-node cluster

Starting the cluster is performed in two steps.

1. We begin with starting the HDFS daemons: the NameNode daemon is started on `master`, and DataNode daemons are started on all slaves (here: `master` and `slave`).
2. Then we start the MapReduce daemons: the JobTracker is started on `master`, and TaskTracker daemons are started on all slaves (here: `master` and `slave`).

HDFS daemons

Run the command `bin/start-dfs.sh` on the machine you want the (primary) NameNode to run on. This will bring up HDFS with the NameNode running on the machine you ran the previous command on, and DataNodes on the machines listed in the `conf/slaves` file.

In our case, we will run `bin/start-dfs.sh` on `master`:

Start the HDFS layer

```
hduser@master:/usr/local/hadoop$ bin/start-dfs.sh
starting namenode, logging to /usr/local/hadoop/bin/../logs/hadoop-hduser-namenode-master.out
slave: Ubuntu 10.04
slave: starting datanode, logging to /usr/local/hadoop/bin/../logs/hadoop-hduser-datanode-slave.out
master: starting datanode, logging to /usr/local/hadoop/bin/../logs/hadoop-hduser-datanode-master.out
master: starting secondarynamenode, logging to /usr/local/hadoop/bin/../logs/hadoop-hduser-secondarynamenode-master.out
hduser@master:/usr/local/hadoop$
```

On `slave`, you can examine the success or failure of this command by inspecting the log file `logs/hadoop-hduser-datanode-slave.log`.

Example output:

```
... INFO org.apache.hadoop.dfs.Storage: Storage directory /app/hadoop/tmp/dfs/data is not formatted.
... INFO org.apache.hadoop.dfs.Storage: Formatting ...
... INFO org.apache.hadoop.dfs.DataNode: Opened server at 50010
... INFO org.mortbay.util.Credential: Checking Resource aliases
... INFO org.mortbay.http.HttpServer: Version Jetty/5.1.4
... INFO org.mortbay.util.Container: Started org.mortbay.jetty.servlet.WebApplicationHandler@17a8a02
... INFO org.mortbay.util.Container: Started WebApplicationContext[/,/]
... INFO org.mortbay.util.Container: Started HttpContext[/logs,/logs]
... INFO org.mortbay.util.Container: Started HttpContext[/static,/static]
... INFO org.mortbay.http.SocketListener: Started SocketListener on 0.0.0.0:50075
... INFO org.mortbay.util.Container: Started org.mortbay.jetty.Server@56a499
... INFO org.apache.hadoop.dfs.DataNode: Starting DataNode in: FSDataset{dirpath='/app/hadoop/tmp/dfs/data/current'}
... INFO org.apache.hadoop.dfs.DataNode: using BLOCKREPORT_INTERVAL of 3538203msec
```

As you can see in `slave`'s output above, it will automatically format its storage directory (specified by the `dfs.data.dir` parameter) if it is not formatted already. It will also create the directory if it does not exist yet.

At this point, the following Java processes should run on `master`…

Java processes on master after starting HDFS daemons

```
hduser@master:/usr/local/hadoop$ jps
14799 NameNode
15314 Jps
14880 DataNode
14977 SecondaryNameNode
hduser@master:/usr/local/hadoop$
```

(the process IDs don't matter of course)

…and the following on `slave`.

Java processes on slave after starting HDFS daemons

```
hduser@slave:/usr/local/hadoop$ jps
15183 DataNode
15616 Jps
hduser@slave:/usr/local/hadoop$
```

MapReduce daemons

Run the command `bin/start-mapred.sh` on the machine you want the JobTracker to run on. This will bring up the MapReduce cluster with the JobTracker running on the machine you ran the previous command on, and TaskTrackers on the machines listed in the `conf/slaves` file.

In our case, we will run `bin/start-mapred.sh` on `master`:

Start the MapReduce layer

```
hduser@master:/usr/local/hadoop$ bin/start-mapred.sh
starting jobtracker, logging to /usr/local/hadoop/bin/../logs/hadoop-hadoop-jobtracker-master.out
slave: Ubuntu 10.04
slave: starting tasktracker, logging to /usr/local/hadoop/bin/../logs/hadoop-hduser-tasktracker-slave.out
master: starting tasktracker, logging to /usr/local/hadoop/bin/../logs/hadoop-hduser-tasktracker-master.out
hduser@master:/usr/local/hadoop$
```

On `slave`, you can examine the success or failure of this command by inspecting the log file `logs/hadoop-hduser-tasktracker-slave.log`. Example output:

```
... INFO org.mortbay.util.Credential: Checking Resource aliases
... INFO org.mortbay.http.HttpServer: Version Jetty/5.1.4
... INFO org.mortbay.util.Container: Started org.mortbay.jetty.servlet.WebApplicationHandler@d19bc8
... INFO org.mortbay.util.Container: Started WebApplicationContext[/,/]
... INFO org.mortbay.util.Container: Started HttpContext[/logs,/logs]
... INFO org.mortbay.util.Container: Started HttpContext[/static,/static]
... INFO org.mortbay.http.SocketListener: Started SocketListener on 0.0.0.0:50060
... INFO org.mortbay.util.Container: Started org.mortbay.jetty.Server@1e63e3d
... INFO org.apache.hadoop.ipc.Server: IPC Server listener on 50050: starting
... INFO org.apache.hadoop.ipc.Server: IPC Server handler 0 on 50050: starting
... INFO org.apache.hadoop.mapred.TaskTracker: TaskTracker up at: 50050
... INFO org.apache.hadoop.mapred.TaskTracker: Starting tracker tracker_slave:50050
... INFO org.apache.hadoop.ipc.Server: IPC Server handler 1 on 50050: starting
... INFO org.apache.hadoop.mapred.TaskTracker: Starting thread: Map-events fetcher for all reduce tasks on tracker_slave:50050
```

At this point, the following Java processes should run on `master`…

Java processes on master after starting MapReduce daemons

```
hduser@master:/usr/local/hadoop$ jps
16017 Jps
14799 NameNode
15686 TaskTracker
14880 DataNode
15596 JobTracker
14977 SecondaryNameNode
hduser@master:/usr/local/hadoop$
```

(the process IDs don't matter of course)

…and the following on `slave`.

Java processes on slave after starting MapReduce daemons

```
hduser@slave:/usr/local/hadoop$ jps
15183 DataNode
15897 TaskTracker
16284 Jps
hduser@slave:/usr/local/hadoop$
```

Stopping the multi-node cluster

Like starting the cluster, stopping it is done in two steps. The workflow however is the opposite of starting.

1. We begin with stopping the MapReduce daemons: the JobTracker is stopped on `master`, and TaskTracker daemons are stopped on all slaves (here: `master` and `slave`).
2. Then we stop the HDFS daemons: the NameNode daemon is stopped on `master`, and DataNode daemons are stopped on all slaves (here: `master` and `slave`).

MapReduce daemons

Run the command `bin/stop-mapred.sh` on the JobTracker machine. This will shut down the MapReduce cluster by stopping the JobTracker daemon running on the machine you ran the previous command on, and TaskTrackers on the machines listed in the `conf/slaves` file.

In our case, we will run `bin/stop-mapred.sh` on `master`:

Stopping the MapReduce layer

```
hduser@master:/usr/local/hadoop$ bin/stop-mapred.sh
stopping jobtracker
slave: Ubuntu 10.04
master: stopping tasktracker
slave: stopping tasktracker
hduser@master:/usr/local/hadoop$
```

Note: The output above might suggest that the JobTracker was running and stopped on "slave", but you can be assured that the JobTracker ran on "master".

At this point, the following Java processes should run on `master`…

Java processes on master after stopping MapReduce daemons

```
hduser@master:/usr/local/hadoop$ jps
14799 NameNode
18386 Jps
14880 DataNode
14977 SecondaryNameNode
hduser@master:/usr/local/hadoop$
```

…and the following on `slave`.

Java processes on slave after stopping MapReduce daemons

```
hduser@slave:/usr/local/hadoop$ jps
15183 DataNode
18636 Jps
hduser@slave:/usr/local/hadoop$
```

HDFS daemons

Run the command `bin/stop-dfs.sh` on the NameNode machine. This will shut down HDFS by stopping the NameNode daemon running on the machine you ran the previous command on, and DataNodes on the machines listed in the `conf/slaves` file.

In our case, we will run `bin/stop-dfs.sh` on `master`:

Stopping the HDFS layer

```
hduser@master:/usr/local/hadoop$ bin/stop-dfs.sh
stopping namenode
slave: Ubuntu 10.04
slave: stopping datanode
master: stopping datanode
master: stopping secondarynamenode
hduser@master:/usr/local/hadoop$
```

(again, the output above might suggest that the NameNode was running and stopped on `slave`, but you can be assured that the NameNode ran on `master`)

At this point, the only following Java processes should run on `master`…

Java processes on master after stopping HDFS daemons

```
hduser@master:/usr/local/hadoop$ jps
18670 Jps
hduser@master:/usr/local/hadoop$
```

…and the following on `slave`.

Java processes on slave after stopping HDFS daemons

```
hduser@slave:/usr/local/hadoop$ jps
18894 Jps
hduser@slave:/usr/local/hadoop$
```

Running a MapReduce job

Just follow the steps described in the section Running a MapReduce job of the single-node cluster tutorial.

I recommend however that you use a larger set of input data so that Hadoop will start several Map and Reduce tasks, and in particular, on *both* `master` and `slave`. After all this installation and configuration work, we want to see the job processed by all machines in the cluster, don't we?

Here's the example input data I have used for the multi-node cluster setup described in this tutorial. I added four more Project Gutenberg etexts to the initial three documents mentioned in the single-node cluster tutorial. All etexts should be in plain text us-ascii encoding.

- The Outline of Science, Vol. 1 (of 4) by J. Arthur Thomson
- The Notebooks of Leonardo Da Vinci
- Ulysses by James Joyce
- The Art of War by 6th cent. B.C. Sunzi
- The Adventures of Sherlock Holmes by Sir Arthur Conan Doyle
- The Devil's Dictionary by Ambrose Bierce
- Encyclopaedia Britannica, 11th Edition, Volume 4, Part 3

Download these etexts, copy them to HDFS, run the WordCount example MapReduce job on `master`, and retrieve the job result from HDFS to your local filesystem.

Here's the example output on `master`… after executing the MapReduce job…

```
hduser@master:/usr/local/hadoop$ bin/hadoop jar hadoop*examples*.jar wordcount  /user/hduser/gutenberg /user/hduser/gutenberg-outp
... INFO mapred.FileInputFormat: Total input paths to process : 7
... INFO mapred.JobClient: Running job: job_0001
... INFO mapred.JobClient:  map 0% reduce 0%
... INFO mapred.JobClient:  map 28% reduce 0%
... INFO mapred.JobClient:  map 57% reduce 0%
... INFO mapred.JobClient:  map 71% reduce 0%
... INFO mapred.JobClient:  map 100% reduce 9%
... INFO mapred.JobClient:  map 100% reduce 68%
... INFO mapred.JobClient:  map 100% reduce 100%
.... INFO mapred.JobClient: Job complete: job_0001
... INFO mapred.JobClient: Counters: 11
... INFO mapred.JobClient:   org.apache.hadoop.examples.WordCount$Counter
... INFO mapred.JobClient:     WORDS=1173099
... INFO mapred.JobClient:     VALUES=1368295
... INFO mapred.JobClient:   Map-Reduce Framework
... INFO mapred.JobClient:     Map input records=136582
... INFO mapred.JobClient:     Map output records=1173099
... INFO mapred.JobClient:     Map input bytes=6925391
... INFO mapred.JobClient:     Map output bytes=11403568
... INFO mapred.JobClient:     Combine input records=1173099
... INFO mapred.JobClient:     Combine output records=195196
... INFO mapred.JobClient:     Reduce input groups=131275
... INFO mapred.JobClient:     Reduce input records=195196
... INFO mapred.JobClient:     Reduce output records=131275
hduser@master:/usr/local/hadoop$
```

…and the logging output on `slave` for its DataNode daemon…

logs/hadoop-hduser-datanode-slave.log (on slave)

```
... INFO org.apache.hadoop.dfs.DataNode: Received block blk_5693969390309798974 from  /192.168.0.1
... INFO org.apache.hadoop.dfs.DataNode: Received block blk_7671491277162757352 from /192.168.0.1
<snipp>
... INFO org.apache.hadoop.dfs.DataNode: Served block blk_-7112133651100166921 to /192.168.0.2
... INFO org.apache.hadoop.dfs.DataNode: Served block blk_-7545080504225510279 to /192.168.0.2
... INFO org.apache.hadoop.dfs.DataNode: Served block blk_-4114464184254609514 to /192.168.0.2
... INFO org.apache.hadoop.dfs.DataNode: Served block blk_-4561652742730019659 to /192.168.0.2
<snipp>
... INFO org.apache.hadoop.dfs.DataNode: Received block blk_-2075170214887808716 from /192.168.0.2 and mirrored to /192.168.0.1:50
... INFO org.apache.hadoop.dfs.DataNode: Received block blk_1422409522782401364 from /192.168.0.2 and mirrored to /192.168.0.1:500
... INFO org.apache.hadoop.dfs.DataNode: Deleting block blk_-2942401177672711226 file /app/hadoop/tmp/dfs/data/current/blk_-29424
... INFO org.apache.hadoop.dfs.DataNode: Deleting block blk_-3019298164878756077 file /app/hadoop/tmp/dfs/data/current/blk_-30192
```

…and on `slave` for its TaskTracker daemon.

logs/hadoop-hduser-tasktracker-slave.log (on slave)

```
... INFO org.apache.hadoop.mapred.TaskTracker: LaunchTaskAction: task_0001_m_000000_0
... INFO org.apache.hadoop.mapred.TaskTracker: LaunchTaskAction: task_0001_m_000001_0
... task_0001_m_000001_0 0.08362164% hdfs://master:54310/user/hduser/gutenberg/ulyss12.txt:0+1561677
... task_0001_m_000000_0 0.07951202% hdfs://master:54310/user/hduser/gutenberg/19699.txt:0+1945731
<snipp>
... task_0001_m_000001_0 0.35611463% hdfs://master:54310/user/hduser/gutenberg/ulyss12.txt:0+1561677
... Task task_0001_m_000001_0 is done.
... task_0001_m_000000_0 1.0% hdfs://master:54310/user/hduser/gutenberg/19699.txt:0+1945731
... LaunchTaskAction: task_0001_m_000006_0
... LaunchTaskAction: task_0001_r_000000_0
... task_0001_m_000000_0 1.0% hdfs://master:54310/user/hduser/gutenberg/19699.txt:0+1945731
... Task task_0001_m_000000_0 is done.
... task_0001_m_000006_0 0.6844295% hdfs://master:54310/user/hduser/gutenberg/132.txt:0+343695
... task_0001_r_000000_0 0.095238104% reduce > copy (2 of 7 at 1.68 MB/s) >
... task_0001_m_000006_0 1.0% hdfs://master:54310/user/hduser/gutenberg/132.txt:0+343695
... Task task_0001_m_000006_0 is done.
... task_0001_r_000000_0 0.14285716% reduce > copy (3 of 7 at 1.02 MB/s) >
<snipp>
... task_0001_r_000000_0 0.14285716% reduce > copy (3 of 7 at 1.02 MB/s) >
... task_0001_r_000000_0 0.23809525% reduce > copy (5 of 7 at 0.32 MB/s) >
... task_0001_r_000000_0 0.6859089% reduce > reduce
... task_0001_r_000000_0 0.7897389% reduce > reduce
... task_0001_r_000000_0 0.86783284% reduce > reduce
... Task task_0001_r_000000_0 is done.
... Received 'KillJobAction' for job: job_0001
... task_0001_r_000000_0 done; removing files.
... task_0001_m_000000_0 done; removing files.
... task_0001_m_000006_0 done; removing files.
... task_0001_m_000001_0 done; removing files.
```

If you want to inspect the job's output data, you need to retrieve the job results from HDFS to your local file system (see instructions in the single-node cluster tutorial.

Caveats

java.io.IOException: Incompatible namespaceIDs

If you observe the error "java.io.IOException: Incompatible namespaceIDs" in the logs of a DataNode (`logs/hadoop-hduser-datanode-.log`), chances are you are affected by issue HDFS-107 (formerly known as HADOOP-1212).

The full error looked like this on my machines:

```
... ERROR org.apache.hadoop.dfs.DataNode: java.io.IOException: Incompatible namespaceIDs in /app/hadoop/tmp/dfs/data: namenode na
        at org.apache.hadoop.dfs.DataStorage.doTransition(DataStorage.java:281)
        at org.apache.hadoop.dfs.DataStorage.recoverTransitionRead(DataStorage.java:121)
        at org.apache.hadoop.dfs.DataNode.startDataNode(DataNode.java:230)
        at org.apache.hadoop.dfs.DataNode.(DataNode.java:199)
        at org.apache.hadoop.dfs.DataNode.makeInstance(DataNode.java:1202)
        at org.apache.hadoop.dfs.DataNode.run(DataNode.java:1146)
        at org.apache.hadoop.dfs.DataNode.createDataNode(DataNode.java:1167)
        at org.apache.hadoop.dfs.DataNode.main(DataNode.java:1326)
```

There are basically two solutions to fix this error as I will describe below.

Solution 1: Start from scratch

This step fixes the problem at the cost of erasing all existing data in the cluster's HDFS file system.

1. Stop the full cluster, i.e. both MapReduce and HDFS layers.
2. Delete the data directory on the problematic DataNode: the directory is specified by `dfs.data.dir` in `conf/hdfs-site.xml`; if you followed this tutorial, the relevant directory is `/app/hadoop/tmp/dfs/data`.
3. Reformat the NameNode. **WARNING: all HDFS data is lost during this process!**
4. Restart the cluster.

When deleting all the HDFS data and starting from scratch does not sound like a good idea (it might be ok during the initial setup/testing), you might give the second approach a try.

Solution 2: Manually update the namespaceID of problematic DataNodes

Big thanks to Jared Stehler for the following suggestion. This workaround is "minimally invasive" as you only have to edit a single file on the problematic DataNodes:

1. Stop the problematic DataNode(s).
2. Edit the value of `namespaceID` in `${dfs.data.dir}/current/VERSION` to match the corresponding value of the current NameNode in `${dfs.name.dir}/current/VERSION`.
3. Restart the fixed DataNode(s).

If you followed the instructions in my tutorials, the full paths of the relevant files are:

- NameNode: `/app/hadoop/tmp/dfs/name/current/VERSION`
- DataNode: `/app/hadoop/tmp/dfs/data/current/VERSION` (background: `dfs.data.dir` is by default set to `${hadoop.tmp.dir}/dfs/data`, and we set `hadoop.tmp.dir` in this tutorial to `/app/hadoop/tmp`).

If you wonder how the contents of `VERSION` look like, here's one of mine:

contents of current/VERSION

```
namespaceID=393514426
storageID=DS-1706792599-10.10.10.1-50010-1204306713481
cTime=1215607609074
storageType=DATA_NODE
layoutVersion=-13
```

Where to go from here

If you're feeling comfortable, you can continue your Hadoop experience with my tutorial on how to code a simple MapReduce job in the Python programming language which can serve as the basis for writing your own MapReduce programs.

Related Links

From yours truly:

- Running Hadoop On Ubuntu Linux (Single-Node Cluster)
- Writing An Hadoop MapReduce Program In Python
- Benchmarking and Stress Testing an Hadoop Cluster with TeraSort, TestDFSIO & Co.

From other people:

- How to debug MapReduce programs
- Hadoop API Overview (Hadoop 2.x)
- Bug HDFS-107: DataNodes should be formatted when the NameNode is formatted
- Bug MAPREDUCE-63: TaskTracker falls into an infinite loop during `reduce > copy` step

Change Log

Only major changes are listed here.

- 2011-07-17: Renamed the Hadoop user from `hadoop` to `hduser` based on readers' feedback. This should make the distinction between the local Hadoop user (now `hduser`), the local Hadoop group (`hadoop`), and the Hadoop CLI tool (`hadoop`) more clear.

Tweet 1

Comments

374 comments ★ 5

Leave a message...

Best Community Share ⚙

Avatar ivan • 10 months ago

Thanks for your tutorial. Here I got a problem.
If I put only one .txt file into dfs, the MapReduce will be completed without any problems.
But when I put more than one file, the job seems pending with below info:

```
" 12/12/08 15:51:28 INFO mapred.JobClient: map 0% reduce 0%
12/12/08 15:51:42 INFO mapred.JobClient: map 50% reduce 0%
12/12/08 15:51:43 INFO mapred.JobClient: map 100% reduce 0%
12/12/08 15:51:51 INFO mapred.JobClient: map 100% reduce 16% "
```

In the same time, output in hadoop-hduser-tasktracker-xxx.log is " ... reduce > copy (1 of 2 at 0.13MB/s)"

20 Reply Share ›

Avatar Nagarjuna Vissa • 6 months ago

How to install hadoop on 1000 nodes at a time. Is there any script to do?

Thanks in advance.

8 Reply Share ›

julion → Nagarjuna Vissa • 2 months ago

Try cssh. You use it in the same way as ssh only this time you execute the same commands on all machines u ssh into
so :
cssh 10.10.10.1 10.10.10.2 .... until 10.10.10.1000
then u simply execute the same commands to install hadoop on one machine and it will take care of the rest
This is assuming they are all the same operating system and have the same filesystem

1 Reply Share ›

Avatar Navie • a year ago

Thank you for your tutorial. That's the first long tutorial for Ubuntu that works perfectly for me :)
btw, I got a warning message "$HADOOP_HOME is depreciated", could you help me with that? My hadoop folder is at /home/hduser/hadoop.

8 Reply Share ›

Michael G. Noll → Navie • a year ago

@Navie: As of this time the warning "$HADOOP_HOME is depreciated" is harmless. See a previous reply of mine for more information on how to fix it anyways if you care.

1 Reply Share ›

Avatar Sudheer • 9 months ago

hi

i followed the above procedure for running the multi node cluster in hadoop.

I followed the above procedure for running the multi node cluster in hadoop,
i run the jps command in master node it is showing only the following it is not showing the namenode
Jps
DataNode
SecondaryNameNode
after this again i reinstall the hadoop and now it is showing along with namenode after running the ./start-dfs.sh
when i run the mapreduce start-mapred.sh then i run the jps command then it is not showing the namenode
then again i reinstall the hadoop and then at this step it is not showing the datanode

can please help me

thanks for the help....

7 ^ | v Reply Share ›

Felipe → Sudheer • 2 months ago

check your Datanode log, maybe you are facing java.io.IOException: Incompatible namespaceIDs

1 ^ | v Reply Share ›

Avatar Sundeep • a year ago

Hi Michael ,
I have followed the tutorial and created a cluster with two nodes. But when I try to run word count program. Its stopping near reduce phase. It does not proceed further. I waited for 15-20 mins and the finally quit the job. I tried many times but there no change every time its getting stuck at reduce phase. Can you suggest me what I need to do.

Regards
Sundeep

6 ^ | v Reply Share ›

Avatar Shuvrow • 9 months ago

Thanks Michael for your nice tutorial about hadoop. I followed your tutorial to setup hadoop in 2 pc but when i try to run word count program, Although it mapped 100%, it stop reducing after 16% or 19% completion. What can i do to complete this word count job?
Thanks
Shuvrow

4 ^ | v Reply Share ›

Avatar Cicci Derpina Ebbasta • 5 months ago

I ufficially love you.Sorry for the openness :D

3 ^ | v Reply Share ›

Avatar Subrata • 9 months ago

Hi Michael,
Your answer to SiD on safe mode issue.
@sid: The NameNode is in Safe Mode — and this is not “a problem”. This is a normal step during the startup of the NameNode.

I followed the same with 3 nodes(1 Master, 2 Slaves). When I ran bin/start-dfs.sh and bin/start-mapred.sh all processes came up. But when i tried to run a example program, got the same error reported by @SID.
And waited for several time but no luck.
For reference I tried to execute below one-
bin/hadoop jar hadoop-examples-1.0.4.jar pi 10 100
Any suggession please going wrong here. @@When tried with single node cluster it worked fine withought any issue.
Thanks
SB.

2 ^ | v Reply Share ›

Avatar varun • 7 months ago

Thanks a ton for making the hadoop tutorial.

I am wondering can the hadoop user in master and slave can be different (i.e on master userhd and in slave hduser).

1 ^ | v Reply Share ›

Avatar Walter • 9 months ago

Hi, the tutorial is excelent, you helped me getting started with this complex mather.

A small detail, could it be that in the "Workaround 1: Start from scratch" instead of looking for the directory in the file called "conf/hdfs-site.xml", it is the directory in the file "conf/core-site.xml" ? Because in the hdfs-site.xml, as you explained, there are the amount of replicas for a block. Thanks again!

1 ^ | v Reply Share ›

Avatar Tinniam V Ganesh • 10 months ago

Hi,
Your posts on SIngle node & multi-node are really great.. However, I am getting the following error - master: ssh: connect to host master port 22: No route to host. I followed all your steps. Unable to figure out the reason. Please help.

Thanks
Ganesh

1 Reply Share ›

Avatar

Guy Rapaport • 14 days ago

A silly question, but it might matter: running the wordcount on the 7 books, my output folder turns out like this:

```
./part-r-00001
./part-r-00000
./_logs
./_logs/history
./_logs/history/job_201309121257_0002_conf.xml
./_logs/history/job_201309121257_0002_1378983973354_hduser_word+count
./_SUCCESS
```

Is it OK that the output is split into 2 files? It seems that there is no intersection between the words in each file, but I just wanted to make sure that it's OK...
Thanks!

Reply Share ›

Michael G. Noll Owner → Guy Rapaport • 13 days ago

@Guy Rapaport: Yes, it is ok if the output is split into multiple files (actually, having just a single file is the exception).

Reply Share ›

Guy Rapaport → Michael G. Noll • 11 days ago

Thanks!

Another question, if I may:

I've performed your tutorial, then continued on to create a cluster of 5 machines
(namenode, jobtracker, and 3 other machines - slave0, slave1, slave2).

It seems that all the machines should know each other - i.e. the slaves must have each other defined in the /etc/hosts file in order for the MapReduce wordcount job to succeed. However, I did not define any ssh password-less communication between them (like I did from the namenode and the jobtracker to each of the slaves).

How is that so? Do the datanodes/tasktrackers communicate directly with each other, in a non-secured protocol, as they do not share any keys?

Reply Share ›

Michael G. Noll Owner → Guy Rapaport • 9 days ago

@Guy Rapaport: You do not need SSH access between the slave machines (only from the master to each slave). The Hadoop daemons use their own protocols and network ports for communication, just like you suspected.

Reply Share ›

Avatar

Rajat Sharma • a month ago

hi Micheal

Great doc you made it simple

thanks

Rajat Sharma

Reply Share ›

Avatar

Rajat Sharma • a month ago

great tutorial Michael

Thanks a lot

Reply Share ›

Avatar

Nayan • a month ago

Its a great tutorial for newbies like me. I have started configuring Hadoop 2.1.0-beta version for single node. Every thing I did and configured well. As a result of JPS, I got that NameNode, DataNode, Secondary NameNode started fine. Then I found out that there is no start-mapred.sh script. So I tried starting the jobtracker using hadoop-daemons.sh (hadoop-daemon.sh --config /home/nayan/dev/hadoop/etc/hadoop/ start jobtracker) and it resulted in failure with message "Sorry, the jobtracker command is no longer supported. You may find similar functionality with the "yarn" shell command.". I do not know what all configuration changes (if any) I need to make. I made changes in "yarn-site.xml" file, as suggested in Hadoop:The Definitive Guide. But could not proceed further. What changes do I need to make?

Thanks,
Nayan

Reply Share ›

Nayan → Nayan • 24 days ago

I followed instructions in Hadoop : The Definitive Guide, and made changes in "yarn-site.xml". Now I am able to start all components. But when I am trying to create a directory in HDFS using command "hdfs dfs -mkdir /usr/hadoop/sample", I am getting error "mkdir: `/home/nayan/dev/hadoop/sample': No such file or directory". But when I tried to the same with "/tmp/hdfs/sample", directories got created. Then I tried to copy files from local system to hdfs directory "hdfs dfs -copyFromLocal /home/nayan/hdfs-samples/*.* /tmp/hdfs/sample/" and I got error :-

13/08/30 17:28:29 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable copyFromLocal: unexpected URISyntaxException

There were 8 files in my local system out of them 7 got imported. For the specific file that missed from being imported, I tried to import that individually.

hdfs dfs -copyFromLocal /home/nayan/hdfs-samples/example.pdf /tmp/hdfs/sample/

It got imported without any issues. I tried the same thing with multiple directories. Every time one file does not imported. I am not sure why this is happening.

Reply Share ›

Nayan ➔ Nayan • a month ago

Has anyone faced this issue???

Reply Share ›

Avatar

Nayan • a month ago

Thanks for the tutorial. It is quite helpful for beginners like me. But I am facing a lot of difficulties while using Hadoop 2.1.0-beta version. It does not have start-mapred.sh, jobtracker commanf is no longer supported. Similar kind of other issues. Where should I look for the updates?

Thanks,
Nayan

Reply Share ›

Avatar

Pankaj Kumar • 2 months ago

Hi Micheal. I created a Blog entry with details how to configure 3 Virtual machines on a single laptop to create the cluster. This will help the readers who do not have more than one machine.
Kindly go through it and if you find it will be useful to readers then include the link in your tutorial.
http://hadoopmagic.wordpress.c...

Reply Share ›

Avatar

Deepak Jha • 4 months ago

Thanks for your Tutorial.

I have installed a two cluster node using two virtual instances of Ubuntu-13.04 and one as a master node and other as a slave node.

The Various hadoop Daemons are running properly on both the Virtual Os but when i am running wordcount jar file then it fails and a log file is created in the output directory stating that:-

```
Job JOBID="job_201306071449_0005" JOBNAME="word count" USER="root" SUBMIT_TIME="1370601324238" JOBCONF="hdfs://master:54310/app/hadoop/tmp/mapred
Job JOBID="job_201306071449_0005" JOB_PRIORITY="NORMAL" .
Job JOBID="job_201306071449_0005" FINISH_TIME="1370601396277" JOB_STATUS="FAILED" FINISHED_MAPS="0" FINISHED_REDUCES="0" FAIL_REASON="Job initial
java\.net\.UnknownHostException: Hadoop_Slave_01 is not a valid Inet address
```

Reply Share ›

Michael G. Noll Owner ➔ Deepak Jha • 3 months ago

@Deepak Jha:

You have set one of your hostnames to an illegal name:

> java.net.UnknownHostException: Hadoop_Slave_01 is not a valid Inet address

Please read the error logs carefully -- you could have spotted this problem easily by yourself.

Reply Share ›

Avatar

Nitin Jain • 4 months ago

Thanks Michael. Just setup a multi-node hadoop cluster with 6 machines!

Reply Share ›

Avatar

Siddiqui Zeeshan • 4 months ago

2013-05-31 00:37:01,970 ERROR org.apache.hadoop.mapred.TaskTracker: Can not start task tracker because java.lang.IllegalArgumentException: Does not contain a valid host:port authority: local

I am getting this error my files are as follows

```
hduser@ubuntu1:/usr/local/hadoop/conf$ cat core-site.xml
```

```
<configuration>
<property>
<name>hadoop.tmp.dir</name>
<value>/app/hadoop/tmp</value>
<description>A base for other temporary directories.</description>
</property>
<property>
<name>fs.default.name</name>
<value>hdfs://ubuntu1:9100</value>
```

see more

^ | ˅ Reply Share ›

Michael G. Noll Owner → Siddiqui Zeeshan • 4 months ago

@Siddiqui Zeeshan: Your XML snippets have been mangled by Disqus' comment system. I tried to salvage as much of your original comment and stored it at https://gist.github.com/miguno....

From what I can tell your mapred-site.xml file is missing the <configuration> tag.

^ | ˅ Reply Share ›

Siddiqui Zeeshan → Michael G. Noll • 4 months ago

You are right sir and a genius.I figured it also but by that time i had posted

Thanks a lot

^ | ˅ Reply Share ›

Avatar Da Purl • 6 months ago

Hi Micheal,

First of all thank you for the wonderful write up. Wondering if anyone ha attemped to build a cluster on Amazon EC2 - using the Free tier. The ones we get for free are micro instances of 613mb memory. I just installed successfully a single node cluster following your tutorial. However, i am not sure if a cluster is possible. I am aware that I can use Whirr and install the basic cluter, but I do want to attempt to install a cluster from scratch by myself. Am not sure if its possible though.

K

^ | ˅ Reply Share ›

Avatar Bharath • 7 months ago

First I like to thank you for publishing such a wonderful guide for hadoop installation.I have a 5 node cluster .I tried individually setting up 5 single node clusters it was working fine.Now I am trying to make a 5 node cluster .When ,I try to copy files from local directory to hdfs.I was getting this error.

```
copyFromLocal: org.apache.hadoop.hdfs.server.namenode.SafeModeException: Cannot create file/user/root/HadoopEbooks/EncyclopaediaBritannica.txt. N
```

Then ,I used this command to off the safe mode.
/usr/local/hadoop/bin/hadoop dfsadmin -safemode leave

Then when I tried to copy

```
/usr/local/hadoop/bin/hadoop dfs -copyFromLocal /root/Desktop/HadoopEbooks /user/root/HadoopEbooks
```

its showing this exception:

```
13/03/09 18:07:49 WARN hdfs.DFSClient: DataStreamer Exception: org.apache.hadoop.ipc.RemoteException: java.io.IOException: File /user/root/Hadoop
        at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.getAdditionalBlock(FSNamesystem.java:1558)
        at org.apache.hadoop.hdfs.server.namenode.NameNode.addBlock(NameNode.java:696)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:57)
<snipp>
13/03/09 18:07:49 WARN hdfs.DFSClient: Error Recovery for block null bad datanode[0] nodes == null
13/03/09 18:07:49 WARN hdfs.DFSClient: Could not get block locations. Source file "/user/root/HadoopEbooks/HadoopEbooks/The Devil's Dictionary.tx
copyFromLocal: java.io.IOException: File /user/root/HadoopEbooks/HadoopEbooks/The Devil's Dictionary.txt could only be replicated to 0 nodes, ins
</snipp>
```

^ | ˅ Reply Share ›

Guruvarma Avvaru → Bharath • 5 months ago

I have set-up a single node cluster. When I try to '-copyFromLocal', I get the same error, what you mentioned above. Could you please help to resolve this.

Tried the following to solve the error
1) Format the namenode
2) Updated the namesapceID in Datanode version

Trying to decrease the 'Non DFS Used' space. I am unable to identify/view the files which occupy this space. My configuration details given below.

Please let me know how to identify/view the non DFS used files.

```
Configured Capacity: 4497805312 (4.19 GB)
Present Capacity: 147365888 (140.54 MB)
DFS Remaining: 147333120 (140.51 MB)
DFS Used: 32768 (32 KB)
DFS Used%: 0.02%
```

see more

Reply Share ›

Avatar

Aman Jot • 7 months ago

hello sir
i m trying to implement multinode cluster on hadoop. im facing a prob when i give command to copy iles to hdfs it gives me the following output:

```
13/03/06 00:24:07 INFO ipc.Client: Retrying connect to server: master/192.168.159.138:54310. Already tried 0 time(s); retry policy is RetryUpToMa
```

plz do tell what could be the prob.

regards

Reply Share ›

Avatar

Javier Escobar • 7 months ago

Hi. Thank you for the tutorial.

I have a question: I have 4 virtualized ubuntu boxes , each one running on a different core (with a multicore processor). A single one is the master, and the others are the slaves. I downloaded 1GB+ of data from project Gutenberg to analize them with the wordcount example.

When I start the cluster, with start-dfs and start-mapred commands, the system monitor of the host server (in which the ubuntu boxes are virtualized) shows that 4 cores of the processor increments its activity (1 per ubuntu box). Nevertheless, when I start the job, only the core of the master box significately increments its activity. This is a normal behavior? Why in the starting process 4 cores increments its activity, but in the job execution it seems that only the master box is working?

Thank you very much in advance.

Reply Share ›

Javier Ricardo Escobar Avila → Javier Escobar • 6 months ago

[SOLVED]

OMG!! After 2 months I solved my issue. The network configuration seemed to be wrong. My master box, in its hosts file, had the entry "127.0.1.1 'hostname' ". So, the servers that hadoop starts only listened to 1 machine (the master machine), the other machines could not connect to the master. Changing the entry with "192.168.XX.XX 'hostname' " the cluster started to work.

So, the moral is: "I MUST CAREFULLY READ THE LOGS" :P

Reply Share ›

Avatar

Nitesh Chauhan • 8 months ago

this is the best post, it worked perfectly for me :)

Reply Share ›

Avatar

Chris L. • 8 months ago

Thanks for the tutorials, they helped me a lot. I would like to ask about any recommendations for use of a hadoop cluster by multiple people. I would like to help people avoid stepping on each other's toes, like accidentally deleting other's files. So I don't think it's advisable to give out the hduser credentials.

Reply Share ›

Noel → Chris L. • 8 months ago

Yes... I would also like to configure hadoop to run by multiple-users and let them access ONLY their own files. (just like the /tmp files in Unix). Is there any such tutorial that I could followed ? As the one one in
http://titan.softnet.tuc.gr:8082/User:xenia/Page_Title/Hadoop_Cluster_Setup_Tutorial
is not really working right.
Hope to hear from someone and many thanks in advanced.....

Reply Share ›

Chris L. ➔ Chris L. • 8 months ago

Thanks for the quick reply. It sounds like a tutorial on this topic would help many people!

If I understand DFS permissions correctly, anyone with network access to the cluster can submit a job, but jobs are only accepted for users with a directory in DFS (e.g., /user/chris).

The use of an edge node looks useful. For me an edge node is a computer with Hadoop binaries and suitable configuration. I think this will provide easy access (easy bcos there's nothing to install nor configure), without granting people direct access to the cluster.

But I've hit many noob problems already now that I have DFS permissions enabled. The first was not using a wide-open hadoop tmp directory (apparently every newly submitted job gets copied there first). The second was not using a proper setting for the mapreduce job tracker staging area (apparently every user needs a directory there too). I suspect I'll hit more.

Hope you'll consider the next tutorial idea :)

Reply Share ›

Michael G. Noll Owner ➔ Chris L. • 8 months ago

Chris, you are right that handing out hduser credentials is ill-advised. It's like giving the root password of a Linux server to everyone with a user account on the box. Really unpleasant things are bound to happen!

Your question would actually take a pretty long answer. But let me at least give you some basic pointers:

- create separate user/group accounts for your researchers (i.e. do not use hduser)
- enable the HDFS trash feature (to recover accidentally deleted files)
- enable HDFS permissions via "dfs.permissions.enabled"
- enable HDFS space quotas to prevent people from filling up the cluster's storage space, thereby impacting other users; keep in mind that HDFS space quotas come with some gotchas and as I have learned the hard way they also expose you to some rather hidden bugs in the HDFS codebase, depending on the patch level / version of Hadoop you are using (like HDFS space quotas running out of sync with the actual sizes of the files in HDFS)
- use a remote metastore for Hive in multi-user environments (though Hive's permission support is still lacking)
- in general, enable similar data permissions for related tools.
- look at what your distribution ships with; for instance, if you run Cloudera CDH3/4 you can also leverage additional tools such as the Hue UI, which provides role-based access to data/data tools
- set up monitoring for your cluster, notably to catch broken jobs (as a rule of thumb any jobs running for 24+ hours is broken) that block computing resources
- enable the preemption feature of the FairScheduler if your Hadoop cluster is using that scheduler (and if your version of FairScheduler supports preemption, not all do); preemption can take away resources from running jobs of user A and grant those resources to user B

Hope this helps!

Reply Share ›

Noel ➔ Michael G. Noll • 8 months ago

Yes... I would also like to configure hadoop to run by multiple-users and let them access ONLY their own files. (just like the /tmp files in Unix). Is there any such tutorial that I could followed ? As the one one in
http://titan.softnet.tuc.gr:8082/User:xenia/Page_Title/Hadoop_Cluster_Setup_Tutorial
is not really working right.

Hope to hear from someone and many thanks in advanced.....

Reply Share ›

Avatar

Panshul Gupta • 9 months ago

Awesome Tutorial,

As u said:

Adding nodes on the fly are not included in the tutorial YET...

any suggestions by when will you be adding a tutorial for adding nodes on the fly?
or will you be adding it at all?
it is explained in the Apache guide but it is much easier and clear when you explain it in a much better way :)

Regards,

Reply Share ›

Avatar

Raf • 9 months ago

Hi,
Can someone tell me what's the difference between running Hadoop the way mention &
creating a MarReduceJob from AWS console?
I am a bit confused.

Thanks
Raf

Reply Share ›

Avatar Hegazi Abdel Hakim Abdel Rahma • 9 months ago

Solved my many problems ... Thanks so much
Merry Xmas :-)

Reply Share ›

Avatar Manish Gupta • 9 months ago

Excellent article. Thank You.
One small thing I would like to mention - Never use an underscore ("_") in the host names of your linux machines. That will give you error "java.lang.IllegalArgumentException: Does not contain a valid host" after starting the dfs/mapred. This I came to know after good 8-9 hours of struggle.

Reply Share ›

Avatar tushar • 9 months ago

if for a master i have setting like

hdfs://master:9000

should i use same port number for slave like

hdfs://slave:9000

Reply Share ›

Avatar Avinash G A • 9 months ago

this is a lovely article. good job mike.....

Reply Share ›

Avatar souri • 10 months ago

Nicely explained, also refer this YouTube video: Creating Hadoop 2-Node Cluster.

Reply Share ›

Load more comments

ALSO ON MICHAEL G. NOLL WHAT'S THIS?

Building an Hadoop 0.20.x version for HBase 0.90.2
33 comments • 2 years ago
Patrick LEE — Thanks a lot for your contribute.

Reading and Writing Avro Files from the Command Line
2 comments • 5 months ago
Joe Travaglini — Nice writeup - lol at the SC reference

Cookie Monster For XMLHttpRequest
2 comments • 6 years ago
Bojan — Hi Michael, this is great. I was searching for a way to remove cookies from ajax calls.But as i see this is only for FF. Can we make this cross browser …

Using Avro in MapReduce jobs with Hadoop, Pig, Hive
1 comment • 3 months ago
super wesman — I'm trying to use your Java example here, but it refers to the deprecated JobConf class. The "official" ColorCount tutorial on the Avro site …

Subscribe Add Disqus to your site

About Me

I am a researcher and software engineer based in Switzerland, Europe. I work for the .COM and .NET DNS registry operator Verisign as the technical lead of its large-scale computing infrastructure based on the Apache Hadoop stack and as a research affiliate at Verisign Labs. Read more »

Contact

michael@michael-noll.com

Follow Me

@miguno
Blog RSS
GitHub

Recent Posts

- Replephant: Analyzing Hadoop Cluster Usage with Clojure
- Using Avro in MapReduce jobs with Hadoop, Pig, Hive
- Understanding the Internal Message Buffers of Storm
- Installing and Running Graphite via RPM and Supervisord
- Multi-Node Storm Cluster Tutorial Published

Latest Tweets

- Status updating...

Follow @miguno