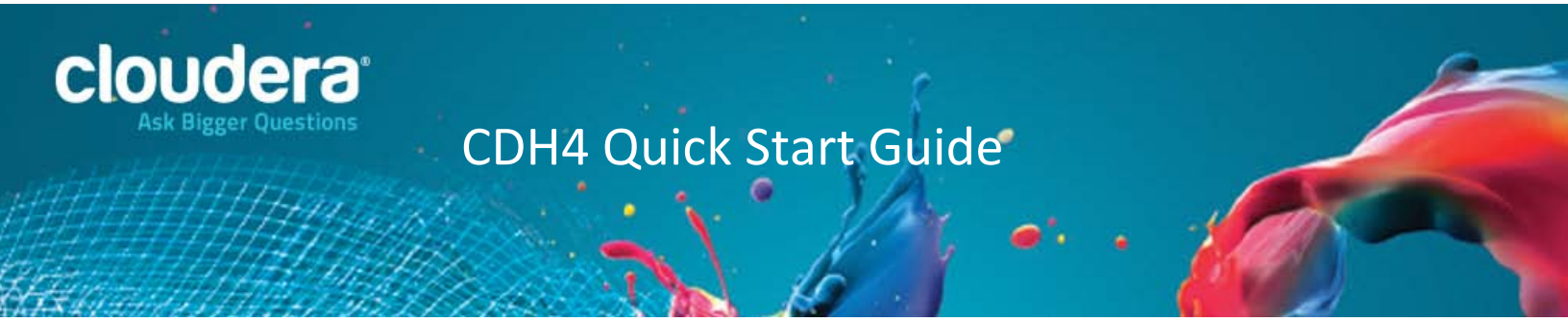

CDH4 Quick Start Guide

Cloudera, Inc.
220 Portage Avenue
Palo Alto, CA 94306
info@cloudera.com
US: 1-888-789-1488
Intl: 1-650-362-0488
www.cloudera.com

Important Notice

Version: CDH4.2

Date: February 24, 2013

Contents

About this Guide

This *CDH4 Quick Start Guide* is for Apache Hadoop developers and system administrators who want to evaluate Cloudera's Distribution Including Apache Hadoop (CDH4). The following sections describe how to quickly install Apache Hadoop and CDH4 components from a Yum, Apt, or zypper/YaST repository on a single Linux node in pseudo-distributed mode:

For more information about installing and configuring CDH4, and deploying in standalone mode and on a cluster, see the *CDH4 Installation Guide* at: https://wiki.cloudera.com/display/DOC/CDH+Installation+Guide.

> **Check out Cloudera Manager Free Edition**
>
> You can use Cloudera Manager Free Edition to automate and simplify many of the steps in a manual installation and deployment of CDH. For more information, see the Cloudera Manager Free Edition documentation.

Before You Install CDH4 on a Single Node

> **Running services:** when starting, stopping and restarting CDH components, always use the `service (8)` command rather than running `/etc/init.d` scripts directly. This is important because `service` sets the current working directory to / and removes most environment variables (passing only `LANG` and `TERM`) so as to create a predictable environment in which to administer the service. If you run the `/etc/init.d` scripts directly, any environment variables you have set remain in force, and could produce unpredictable results. (If you install CDH from packages, `service` will be installed as part of the Linux Standard Base (LSB).)

Before you install CDH4 on a single node, there are some important steps you need to do to prepare your system:

1. Verify you are using a supported operating system for CDH4. See CDH4 Requirements and Supported Versions.
2. If you haven't already done so, install the Oracle Java Development Kit (JDK) before deploying CDH4. See the section below: Install the Oracle Java Development Kit.

   > **Important**
   >
   > On SLES 11 platforms, do not install or try to use the IBM Java version bundled with the SLES distribution; Hadoop will not run correctly with that version. Install the Oracle JDK following directions under Install the Oracle Java Development Kit.

Install the Oracle Java Development Kit

If you have already installed the Oracle JDK, skip this step and proceed to Installing CDH4 on a Single Linux Node in Pseudo-distributed Mode. Install the Oracle Java Development Kit (JDK) before deploying CDH4.

- To install the JDK, follow the instructions under Oracle JDK Installation. The completed installation must meet the requirements in the box below.
- If you have already installed a version of the JDK, make sure your installation meets the requirements in the box below.

Requirements:

- CDH4 requires Oracle JDK 1.6. Cloudera recommends version 1.6.0_31. The minimum supported version is 1.6.0_8.

After installing the JDK, and **before installing and deploying CDH**:

- If you are deploying CDH on a cluster, make sure you have the same version of the Oracle JDK on each node.
- Make sure the `JAVA_HOME` environment variable is set for the root user on each node. You can check by using a command such as

```
$ sudo env | grep JAVA_HOME
```

It should be set to point to the directory where the JDK is installed, as shown in the example below.

You may be able to install the Oracle JDK with your package manager, depending on your choice of operating system.

Oracle JDK Installation

Important

The Oracle JDK installer is available both as an RPM-based installer (note the "`-rpm`" modifier before the `bin` file extension) for RPM-based systems, and as a binary installer for other systems. Make sure you install the `jdk-6uXX-linux-x64-rpm.bin` file for 64-bit systems, or `jdk-6uXX-linux-i586-rpm.bin` for 32-bit systems.

On SLES 11 platforms, do not install or try to use the IBM Java version bundled with the SLES distribution; Hadoop will not run correctly with that version. Install the Oracle JDK by following the instructions below.

To install the Oracle JDK:

1. Download one of the recommended versions of the Oracle JDK from this page, which you can also reach by going to the Java SE Downloads page and clicking on the **Previous Releases** tab and then on the **Java SE 6** link. (These links and directions were correct at the time of writing, but the page is restructured frequently.)
2. Install the Oracle JDK following the directions on the the Java SE Downloads page.
3. As the root user, set `JAVA_HOME` to the directory where the JDK is installed; for example:

```
# export JAVA_HOME=<jdk-install-dir>
# export PATH=$JAVA_HOME/bin:$PATH
```

   where `<jdk-install-dir>` might be something like `/usr/java/jdk1.6.0_31`, depending on the system configuration and where the JDK is actually installed.

Installing CDH4 on a Single Linux Node in Pseudo-distributed Mode

You can evaluate CDH4 by quickly installing Apache Hadoop and CDH4 components on a single Linux node in pseudo-distributed mode. In pseudo-distributed mode, Hadoop processing is distributed over all of the cores/processors on a single machine. Hadoop writes all files to the Hadoop Distributed File System (HDFS), and all services and daemons communicate over local TCP sockets for inter-process communication.

MapReduce 2.0 (YARN)

MapReduce has undergone a complete overhaul and CDH4 now includes MapReduce 2.0 (MRv2). The fundamental idea of MRv2's YARN architecture is to split up the two primary responsibilities of the JobTracker — resource management and job scheduling/monitoring — into separate daemons: a global ResourceManager (RM) and per-application ApplicationMasters (AM).
With MRv2, the ResourceManager (RM) and per-node NodeManagers (NM), form the data-computation framework. The ResourceManager service effectively replaces the functions of the JobTracker, and NodeManagers run on slave nodes instead of TaskTracker daemons. The per-application ApplicationMaster is, in effect, a framework specific library and is tasked with negotiating resources from the ResourceManager and working with the NodeManager(s) to execute and monitor the tasks. For details of the new architecture, see Apache Hadoop NextGen MapReduce (YARN).

> **Note:** Cloudera does not consider the current upstream MRv2 release stable yet, and it could potentially change in non-backwards-compatible ways. Cloudera recommends that you use MRv1 unless you have particular reasons for using MRv2, which should not be considered production-ready.

For more information about the two implementations (MRv1 and MRv2) see the discussion under Apache Hadoop MapReduce in the "What's New in Beta 1" section of New Features in CDH4.

See also Selecting Appropriate JAR files for your MRv1 and YARN Jobs.

Important

For installations in pseudo-distributed mode, there are separate `conf-pseudo` packages for an installation that includes MRv1 (`hadoop-0.20-conf-pseudo`) or an installation that includes YARN (`hadoop-conf-pseudo`). Only one `conf-pseudo` package can be installed at a time: if you want to change from one to the other, you must uninstall the one currently installed.

Installing CDH4 with MRv1 on a Single Linux Node in Pseudo-distributed mode

Important

- **Running services:** when starting, stopping and restarting CDH components, always use the `service (8)` command rather than running `/etc/init.d` scripts directly. This is important because `service` sets the current working directory to / and removes most environment variables (passing only `LANG` and `TERM`) so as to create a predictable environment in which to administer the service. If you run the `/etc/init.d` scripts directly, any environment variables you have set remain in force, and could produce unpredictable results. (If you install CDH from packages, `service` will be installed as part of the Linux Standard Base (LSB).)
- **Java Development Kit:** if you have not already done so, install the Oracle Java Development Kit (JDK) before deploying CDH4. Follow these instructions.

On Red Hat/CentOS/Oracle 5 or Red Hat 6 systems, do the following:

Download the CDH4 Package

1. Click the entry in the table below that matches your Red Hat or CentOS system, choose **Save File**, and save the file to a directory to which you have write access (it can be your home directory).

For OS Version	Click this Link
Red Hat/CentOS/Oracle 5	Red Hat/CentOS/Oracle 5 link
Red Hat/CentOS 6 (32-bit)	Red Hat/CentOS 6 link (32-bit)
Red Hat/CentOS 6 (64-bit)	Red Hat/CentOS 6 link (64-bit)

2. Install the RPM.
   For Red Hat/CentOS/Oracle 5:

```
$ sudo yum --nogpgcheck localinstall cloudera-cdh-4-0.x86_64.rpm
```

   For Red Hat/CentOS 6 (32-bit):

```
$ sudo yum --nogpgcheck localinstall cloudera-cdh-4-0.i386.rpm
```

   For Red Hat/CentOS 6 (64-bit):

```
$ sudo yum --nogpgcheck localinstall cloudera-cdh-4-0.x86_64.rpm
```

> **Note**
>
> For instructions on how to add a CDH4 yum repository or build your own CDH4 yum repository, see Installing CDH4 On Red Hat-compatible systems.

Install CDH4

1. (Optionally) add a repository key. Add the Cloudera Public GPG Key to your repository by executing one of the the following commands:
   - **For Red Hat/CentOS/Oracle 5 systems:**

```
$ sudo rpm --import
http://archive.cloudera.com/cdh4/redhat/5/x86_64/cdh/RPM-GPG-
KEY-cloudera
```

   - **For Red Hat/CentOS 6 systems:**

```
$ sudo rpm --import
http://archive.cloudera.com/cdh4/redhat/6/x86_64/cdh/RPM-GPG-
KEY-cloudera
```

2. Install Hadoop in pseudo-distributed mode:

   To install Hadoop with MRv1:

```
$ sudo yum install hadoop-0.20-conf-pseudo
```

On SLES systems, do the following:

Download and install the CDH4 package

1. Click this link, choose **Save File**, and save it to a directory to which you have write access (it can be your home directory).
2. Install the RPM:

```
$ sudo rpm -i cloudera-cdh-4-0.x86_64.rpm
```

> **Note**
>
> For instructions on how to add a CDH4 SLES repository or build your own CDH4 SLES repository, see Installing CDH4 On SLES systems.

Install CDH4

1. (Optionally) add a repository key. Add the Cloudera Public GPG Key to your repository by executing the following command:
   - **For all SLES systems:**

```
$ sudo rpm --import
http://archive.cloudera.com/cdh4/sles/11/x86_64/cdh/RPM-GPG-
KEY-cloudera
```

2. Install Hadoop in pseudo-distributed mode:

   To install Hadoop with MRv1:

```
$ sudo zypper install hadoop-0.20-conf-pseudo
```

On Ubuntu and other Debian systems, do the following:

Download and install the package

1. Click one of the following:
   this link for a Squeeze system, *or*
   this link for a Lucid system
   this link for a Precise system.

2. Install the package. Do one of the following:
   Choose **Open with** in the download window to use the package manager, *or*
   Choose **Save File**, save the package to a directory to which you have write access (it can be your home directory) and install it from the command line, for example:

```
sudo dpkg -i cdh4-repository_1.0_all.deb
```

> **Note**
>
> For instructions on how to add a CDH4 Debian repository or build your own CDH4 Debian repository, see Installing CDH4 on Ubuntu or Debian systems.

Install CDH4

1. (Optionally) add a repository key. Add the Cloudera Public GPG Key to your repository by executing the following command:
   - **For Ubuntu Lucid systems:**

```
$ curl -s
http://archive.cloudera.com/cdh4/ubuntu/lucid/amd64/cdh/archive
.key | sudo apt-key add -
```

   - **For Ubuntu Precise systems:**

```
$ curl -s
http://archive.cloudera.com/cdh4/ubuntu/precise/amd64/cdh/archi
ve.key | sudo apt-key add -
```

   - **For Debian Squeeze systems:**

```
$ curl -s
http://archive.cloudera.com/cdh4/debian/squeeze/amd64/cdh/archi
ve.key | sudo apt-key add -
```

2. Install Hadoop in pseudo-distributed mode:

   To install Hadoop with MRv1:

```
$ sudo apt-get update
$ sudo apt-get install hadoop-0.20-conf-pseudo
```

Starting Hadoop and Verifying it is Working Properly:

For MRv1, a pseudo-distributed Hadoop installation consists of one node running all five Hadoop daemons: `namenode`, `jobtracker`, `secondarynamenode`, `datanode`, and `tasktracker`.

To verify the `hadoop-0.20-conf-pseudo` packages on your system.

- To view the files on Red Hat or SLES systems:

```
$ rpm -ql hadoop-0.20-conf-pseudo
```

- To view the files on Ubuntu systems:

```
$ dpkg -L hadoop-0.20-conf-pseudo
```

The new configuration is self-contained in the `/etc/hadoop/conf.pseudo.mr1` directory.

> **Note**
>
> The Cloudera packages use the `alternatives` framework for managing which Hadoop configuration is active. All Hadoop components search for the Hadoop configuration in `/etc/hadoop/conf`.

To start Hadoop, proceed as follows.

Step 1: Format the NameNode.

Before starting the NameNode for the first time you **must** format the file system.

```
$ sudo -u hdfs hdfs namenode -format
```

> **Note**
>
> Make sure you perform the format of the NameNode as user `hdfs`. You can do this as part of the command string, using `sudo -u hdfs` as in the command above.

Note

If Kerberos is enabled, do not use commands in the form `sudo -u <user> <command>`; they will fail with a security error. Instead, use the following commands:
`$ kinit <user>` (if you are using a password) *or*
`$ kinit -kt <keytab> <principal>` (if you are using a `keytab`)
and then, for each command executed by this user,
`$ <command>`

Important

In earlier releases, the `hadoop-conf-pseudo` package automatically formatted HDFS on installation. In CDH4, you must do this explicitly.

Step 2: Start HDFS

```
for x in `cd /etc/init.d ; ls hadoop-hdfs-*` ; do sudo service $x start ; done
```

To verify services have started, you can check the web console. The NameNode provides a web console http://localhost:50070/ for viewing your Distributed File System (DFS) capacity, number of DataNodes, and logs. In this pseudo-distributed configuration, you should see one live DataNode named `localhost`.

Step 3: Create the /tmp Directory

Create the `/tmp` directory and set permissions:

Important

If you do not create `/tmp` properly, with the right permissions as shown below, you may have problems with CDH components later. Specifically, if you don't create `/tmp` yourself, another process may create it automatically with restrictive permissions that will prevent your other applications from using it.

Create the `/tmp` directory after HDFS is up and running, and set its permissions to 1777 (`drwxrwxrwt`), as follows:

```
$ sudo -u hdfs hadoop fs -mkdir /tmp
$ sudo -u hdfs hadoop fs -chmod -R 1777 /tmp
```

Step 4: Create the MapReduce system directories:

```
sudo -u hdfs hadoop fs -mkdir -p /var/lib/hadoop-
hdfs/cache/mapred/mapred/staging
sudo -u hdfs hadoop fs -chmod 1777 /var/lib/hadoop-
hdfs/cache/mapred/mapred/staging
sudo -u hdfs hadoop fs -chown -R mapred /var/lib/hadoop-hdfs/cache/mapred
```

Step 5: Verify the HDFS File Structure

```
$ sudo -u hdfs hadoop fs -ls -R /
```

You should see:

```
drwxrwxrwt   - hdfs supergroup          0 2012-04-19 15:14 /tmp
drwxr-xr-x   - hdfs     supergroup          0 2012-04-19 15:16 /var
drwxr-xr-x   - hdfs     supergroup          0 2012-04-19 15:16 /var/lib
drwxr-xr-x   - hdfs     supergroup          0 2012-04-19 15:16
/var/lib/hadoop-hdfs
drwxr-xr-x   - hdfs     supergroup          0 2012-04-19 15:16
/var/lib/hadoop-hdfs/cache
drwxr-xr-x   - mapred   supergroup          0 2012-04-19 15:19
/var/lib/hadoop-hdfs/cache/mapred
drwxr-xr-x   - mapred   supergroup          0 2012-04-19 15:29
/var/lib/hadoop-hdfs/cache/mapred/mapred
drwxrwxrwt   - mapred   supergroup          0 2012-04-19 15:33
/var/lib/hadoop-hdfs/cache/mapred/mapred/staging
```

Step 6: Start MapReduce

```
for x in `cd /etc/init.d ; ls hadoop-0.20-mapreduce-*` ; do sudo service $x
stop ; done
```

To verify services have started, you can check the web console. The `JobTracker` provides a web console http://localhost:50030/ for viewing and running completed and failed jobs with logs.

Step 7: Create User Directories

Create a home directory for each MapReduce user. It is best to do this on the NameNode; for example:

```
$ sudo -u hdfs hadoop fs -mkdir  /user/<user>
$ sudo -u hdfs hadoop fs -chown <user> /user/<user>
```

where <user> is the Linux username of each user.

Alternatively, you can log in as each Linux user (or write a script to do so) and create the home directory as follows:

```
sudo -u hdfs hadoop fs -mkdir /user/$USER
sudo -u hdfs hadoop fs -chown $USER /user/$USER
```

Running an example application with MRv1

1. Create a home directory on HDFS for the user who will be running the job (for example, `joe`):

   ```
   sudo -u hdfs hadoop fs -mkdir /user/joe
   sudo -u hdfs hadoop fs -chown joe /user/joe
   ```

   Do the following steps as the user `joe`.

2. Make a directory in HDFS called `input` and copy some XML files into it by running the following commands:

   ```
   $ hadoop fs -mkdir input
   $ hadoop fs -put /etc/hadoop/conf/*.xml input
   $ hadoop fs -ls input
   Found 3 items:
   -rw-r--r--   1 joe supergroup       1348 2012-02-13 12:21 input/core-
   site.xml
   -rw-r--r--   1 joe supergroup       1913 2012-02-13 12:21 input/hdfs-
   site.xml
   -rw-r--r--   1 joe supergroup       1001 2012-02-13 12:21
   input/mapred-site.xml
   ```

3. Run an example Hadoop job to grep with a regular expression in your input data.

   ```
   $ /usr/bin/hadoop jar /usr/lib/hadoop-0.20-mapreduce/hadoop-
   examples.jar grep input output 'dfs[a-z.]+'
   ```

4. After the job completes, you can find the output in the HDFS directory named `output` because you specified that output directory to Hadoop.

```
$ hadoop fs -ls
Found 2 items
drwxr-xr-x   - joe supergroup  0 2009-08-18 18:36 /user/joe/input
drwxr-xr-x   - joe supergroup  0 2009-08-18 18:38 /user/joe/output
```

You can see that there is a new directory called `output`.

5. List the output files.

```
$ hadoop fs -ls output
Found 2 items
drwxr-xr-x  -  joe supergroup     0 2009-02-25 10:33
/user/joe/output/_logs
-rw-r--r--  1  joe supergroup  1068 2009-02-25 10:33
/user/joe/output/part-00000
-rw-r--r-   1  joe supergroup     0 2009-02-25 10:33
/user/joe/output/_SUCCESS
```

6. Read the results in the output file; for example:

```
$ hadoop fs -cat output/part-00000 | head
1       dfs.datanode.data.dir
1       dfs.namenode.checkpoint.dir
1       dfs.namenode.name.dir
1       dfs.replication
1       dfs.safemode.extension
1       dfs.safemode.min.datanodes
```

Installing CDH4 with YARN on a Single Linux Node in Pseudo-distributed mode

Before you start, uninstall MRv1 if necessary

If you have already installed MRv1 following the steps in the previous section, you now need to uninstall `hadoop-0.20-conf-pseudo` before running YARN. Proceed as follows.

1. Stop the daemons:

```
$ for x in `cd /etc/init.d ; ls hadoop-hdfs-*` ; do sudo service $x
stop ; done
$ for x in 'cd /etc/init.d ; ls hadoop-0.20-mapreduce-* ; do sudo
service $x stop ; done
```

2. Remove `hadoop-0.20-conf-pseudo`:

- On Red Hat-compatible systems:

```
sudo yum remove hadoop-0.20-conf-pseudo hadoop-0.20-mapreduce-*
```

- On SLES systems:

```
sudo zypper remove hadoop-0.20-conf-pseudo hadoop-0.20-mapreduce-*
```

- On Ubuntu or Debian systems:

```
sudo apt-get remove hadoop-0.20-conf-pseudo hadoop-0.20-mapreduce-*
```

Note

In this case (after uninstalling `hadoop-0.20-conf-pseudo`) you can skip the package download steps below.

Important

If you have not already done so, install the Oracle Java Development Kit (JDK) before deploying CDH4. Follow these instructions.

On Red Hat/CentOS/Oracle 5 or Red Hat 6 systems, do the following:

Download the CDH4 Package

1. Click the entry in the table below that matches your Red Hat or CentOS system, choose **Save File**, and save the file to a directory to which you have write access (it can be your home directory).

For OS Version	Click this Link
Red Hat/CentOS/Oracle 5	Red Hat/CentOS/Oracle 5 link
Red Hat/CentOS 6 (32-bit)	Red Hat/CentOS 6 link (32-bit)
Red Hat/CentOS 6 (64-bit)	Red Hat/CentOS 6 link (64-bit)

2. Install the RPM.
   For Red Hat/CentOS/Oracle 5:

```
$ sudo yum --nogpgcheck localinstall cloudera-cdh-4-0.x86_64.rpm
```

   For Red Hat/CentOS 6 (32-bit):

```
$ sudo yum --nogpgcheck localinstall cloudera-cdh-4-0.i386.rpm
```

   For Red Hat/CentOS 6 (64-bit):

```
$ sudo yum --nogpgcheck localinstall cloudera-cdh-4-0.x86_64.rpm
```

Note

For instructions on how to add a CDH4 yum repository or build your own CDH4 yum repository, see Installing CDH4 On Red Hat-compatible systems.

Install CDH4

1. (Optionally) add a repository key. Add the Cloudera Public GPG Key to your repository by executing the following command:
   - **For Red Hat/CentOS/Oracle 5 systems:**

```
$ sudo rpm --import
http://archive.cloudera.com/cdh4/redhat/5/x86_64/cdh/RPM-GPG-
KEY-cloudera
```

   - **For Red Hat/CentOS/Oracle 5 systems:**

```
$ sudo rpm --import
http://archive.cloudera.com/cdh4/redhat/5/x86_64/cdh/RPM-GPG-
KEY-cloudera
```

2. Install Hadoop in pseudo-distributed mode:

   To install Hadoop with YARN:

```
$ sudo yum install hadoop-conf-pseudo
```

On SLES systems, do the following:

Download and install the CDH4 package

1. Click this link, choose **Save File**, and save it to a directory to which you have write access (it can be your home directory).
2. Install the RPM:

```
$ sudo rpm -i cloudera-cdh-4-0.x86_64.rpm
```

> **Note**
>
> For instructions on how to add a CDH4 SLES repository or build your own CDH4 SLES repository, see Installing CDH4 On SLES systems.

Install CDH4

1. (Optionally) add a repository key. Add the Cloudera Public GPG Key to your repository by executing the following command:
   - **For all SLES systems:**

```
$ sudo rpm --import
http://archive.cloudera.com/cdh4/sles/11/x86_64/cdh/RPM-GPG-
KEY-cloudera
```

2. Install Hadoop in pseudo-distributed mode:

   To install Hadoop with YARN:

```
$ sudo zypper install hadoop-conf-pseudo
```

On Ubuntu and other Debian systems, do the following:

Download and install the package

1. Click one of the following:
   this link for a Squeeze system, *or*
   this link for a Lucid system
   this link for a Precise system.
2. Install the package. Do one of the following:
   Choose **Open with** in the download window to use the package manager, *or*

Choose **Save File**, save the package to a directory to which you have write access (it can be your home directory) and install it from the command line, for example:

```
sudo dpkg -i cdh4-repository_1.0_all.deb
```

> **Note**
>
> For instructions on how to add a CDH4 Debian repository or build your own CDH4 Debian repository, see Installing CDH4 On Ubuntu or Debian systems.

Install CDH4

1. (Optionally) add a repository key. Add the Cloudera Public GPG Key to your repository by executing the following command:
   - **For Ubuntu Lucid systems:**

     ```
     $ curl -s
     http://archive.cloudera.com/cdh4/ubuntu/lucid/amd64/cdh/archive
     .key | sudo apt-key add -
     ```

   - **For Ubuntu Precise systems:**

     ```
     $ curl -s
     http://archive.cloudera.com/cdh4/ubuntu/precise/amd64/cdh/archi
     ve.key | sudo apt-key add -
     ```

   - **For Debian Squeeze systems:**

     ```
     $ curl -s
     http://archive.cloudera.com/cdh4/debian/squeeze/amd64/cdh/archi
     ve.key | sudo apt-key add -
     ```

2. Install Hadoop in pseudo-distributed mode:

   To install Hadoop with YARN:

   ```
   $ sudo apt-get update
   $ sudo apt-get install hadoop-conf-pseudo
   ```

Starting Hadoop and Verifying it is Working Properly

For YARN, a pseudo-distributed Hadoop installation consists of one node running all five Hadoop daemons: `namenode`, `secondarynamenode`, `resourcemanager`, `datanode`, and `nodemanager`.

- To view the files on Red Hat or SLES systems:

```
$ rpm -ql hadoop-conf-pseudo
```

- To view the files on Ubuntu systems:

```
$ dpkg -L hadoop-conf-pseudo
```

The new configuration is self-contained in the `/etc/hadoop/conf.pseudo` directory.

> **Note**
>
> The Cloudera packages use the `alternative` framework for managing which Hadoop configuration is active. All Hadoop components search for the Hadoop configuration in `/etc/hadoop/conf`.

To start Hadoop, proceed as follows.

Step 1: Format the NameNode.

Before starting the NameNode for the first time you **must** format the file system.

```
$ sudo -u hdfs hdfs namenode -format
```

> **Note**
>
> Make sure you perform the format of the NameNode as user `hdfs`. You can do this as part of the command string, using `sudo -u hdfs` as in the command above.

> **Important**
>
> In earlier releases, the `hadoop-conf-pseudo` package automatically formatted HDFS on installation. In CDH4, you must do this explicitly.

Step 2: Start HDFS

```
for x in `cd /etc/init.d ; ls hadoop-hdfs-*` ; do sudo service $x start ;
done
```

To verify services have started, you can check the web console. The NameNode provides a web console http://localhost:50070/ for viewing your Distributed File System (DFS) capacity, number of DataNodes, and logs. In this pseudo-distributed configuration, you should see one live DataNode named `localhost`.

Step 3: Create the /tmp Directory

1. Remove the old `/tmp` if it exists:

```
sudo -u hdfs hadoop fs -rm -r /tmp
```

2. Create a new `/tmp` directory and set permissions:

```
sudo -u hdfs hadoop fs -mkdir /tmp
sudo -u hdfs hadoop fs -chmod -R 1777 /tmp
```

Step 4: Create Staging and Log Directories

Create the staging directory and set permissions:

```
sudo -u hdfs hadoop fs -mkdir /tmp/hadoop-yarn/staging
sudo -u hdfs hadoop fs -chmod -R 1777 /tmp/hadoop-yarn/staging
```

Create the `done_intermediate` directory under the staging directory and set permissions:

```
sudo -u hdfs hadoop fs -mkdir /tmp/hadoop-
yarn/staging/history/done_intermediate
sudo -u hdfs hadoop fs -chmod -R 1777 /tmp/hadoop-
yarn/staging/history/done_intermediate
```

Change ownership on the staging directory and subdirectory:

```
sudo -u hdfs hadoop fs -chown -R mapred:mapred /tmp/hadoop-yarn/staging
```

Create the `/var/log/hadoop-yarn` directory and set ownership:

```
sudo -u hdfs hadoop fs -mkdir /var/log/hadoop-yarn
sudo -u hdfs hadoop fs -chown yarn:mapred /var/log/hadoop-yarn
```

You need to create this directory because it is the parent of `/var/log/hadoop-yarn/apps` which is explicitly configured in the `yarn-site.xml`.

Step 5: Verify the HDFS File Structure:

Run the following command:

```
$ sudo -u hdfs hadoop fs -ls -R /
```

You should see the following directory structure:

```
drwxrwxrwt   - hdfs   supergroup          0 2012-05-31 15:31 /tmp
drwxr-xr-x   - hdfs   supergroup          0 2012-05-31 15:31 /tmp/hadoop-yarn
drwxrwxrwt   - mapred mapred              0 2012-05-31 15:31 /tmp/hadoop-
yarn/staging
drwxr-xr-x   - mapred mapred              0 2012-05-31 15:31 /tmp/hadoop-
yarn/staging/history
drwxrwxrwt   - mapred mapred              0 2012-05-31 15:31 /tmp/hadoop-
yarn/staging/history/done_intermediate
drwxr-xr-x   - hdfs   supergroup          0 2012-05-31 15:31 /var
drwxr-xr-x   - hdfs   supergroup          0 2012-05-31 15:31 /var/log
drwxr-xr-x   - yarn   mapred              0 2012-05-31 15:31 /var/log/hadoop-
yarn
```

Step 6: Start YARN

```
sudo service hadoop-yarn-resourcemanager start
sudo service hadoop-yarn-nodemanager start
sudo service hadoop-mapreduce-historyserver start
```

Step 7: Create User Directories

Create a home directory for each MapReduce user. It is best to do this on the NameNode; for example:

```
$ sudo -u hdfs hadoop fs -mkdir  /user/<user>
$ sudo -u hdfs hadoop fs -chown <user> /user/<user>
```

where <user> is the Linux username of each user.

Alternatively, you can log in as each Linux user (or write a script to do so) and create the home directory as follows:

```
sudo -u hdfs hadoop fs -mkdir /user/$USER
sudo -u hdfs hadoop fs -chown $USER /user/$USER
```

Running an example application with YARN

1. Create a home directory on HDFS for the user who will be running the job (for example, `joe`):

   ```
   sudo -u hdfs hadoop fs -mkdir /user/joe
   sudo -u hdfs hadoop fs -chown joe /user/joe
   ```

   Do the following steps as the user `joe`.

2. Make a directory in HDFS called `input` and copy some XML files into it by running the following commands in pseudo-distributed mode:

   ```
   $ hadoop fs -mkdir input
   $ hadoop fs -put /etc/hadoop/conf/*.xml input
   $ hadoop fs -ls input
   Found 3 items:
   -rw-r--r--   1 joe supergroup       1348 2012-02-13 12:21 input/core-
   site.xml
   -rw-r--r--   1 joe supergroup       1913 2012-02-13 12:21 input/hdfs-
   site.xml
   -rw-r--r--   1 joe supergroup       1001 2012-02-13 12:21
   input/mapred-site.xml
   ```

3. Set `HADOOP_MAPRED_HOME` for user `joe`:

   ```
   $ export HADOOP_MAPRED_HOME=/usr/lib/hadoop-mapreduce
   ```

4. Run an example Hadoop job to `grep` with a regular expression in your input data.

   ```
   $ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar
   grep input output23 'dfs[a-z.]+'
   ```

5. After the job completes, you can find the output in the HDFS directory named `output23` because you specified that output directory to Hadoop.

```
$ hadoop fs -ls
Found 2 items
drwxr-xr-x   - joe supergroup  0 2009-08-18 18:36 /user/joe/input
drwxr-xr-x   - joe supergroup  0 2009-08-18 18:38 /user/joe/output23
```

   You can see that there is a new directory called `output23`.

6. List the output files.

```
$ hadoop fs -ls output23
Found 2 items
drwxr-xr-x  -  joe supergroup     0 2009-02-25 10:33
/user/joe/output23/_SUCCESS
-rw-r--r--  1  joe supergroup  1068 2009-02-25 10:33
/user/joe/output23/part-r-00000
```

7. Read the results in the output file.

```
$ hadoop fs -cat output23/part-r-00000 | head
1    dfs.safemode.min.datanodes
1    dfs.safemode.extension
1    dfs.replication
1    dfs.permissions.enabled
1    dfs.namenode.name.dir
1    dfs.namenode.checkpoint.dir
1    dfs.datanode.data.dir
```

Components That Require Additional Configuration

The following CDH components require additional configuration after installation.

- HBase. For more information, see "HBase Installation" in the *CDH4 Installation Guide* at: https://ccp.cloudera.com/display/CDHDOC/HBase+Installation
- ZooKeeper. For more information, see "ZooKeeper Installation" in the *CDH4 Installation Guide* at: https://ccp.cloudera.com/display/CDHDOC/ZooKeeper+Installation
- Snappy. For more information, see "Snappy Installation" in the *CDH4 Installation Guide* at: https://ccp.cloudera.com/display/CDHDOC/Snappy+Installation
- Hue. For more information, see "Hue Installation" in the *CDH4 Installation Guide* at: https://ccp.cloudera.com/display/CDHDOC/Hue+Installation

- Oozie. For more information, see "Oozie Installation" in the *CDH4 Installation Guide* at: https://ccp.cloudera.com/display/CDHDOC/Oozie+Installation

Next Steps

- Learn more about installing and configuring CDH4. See the *CDH4 Installation Guide* at: https://ccp.cloudera.com/display/CDHDOC/CDH4+Installation+Guide
- Learn how to deploy CDH4 in fully-distributed mode on a cluster of machines. See "Deploying CDH4 on a Cluster" at: https://ccp.cloudera.com/display/CDHDOC/Deploying+CDH4+on+a+Cluster
- Watch Cloudera's training videos and work through Cloudera's exercises to learn how to write your first MapReduce job. See **Training videos and exercises** at: http://www.cloudera.com/hadoop-training
- Learn how to quickly and easily use Whirr to run CDH4 clusters on cloud providers' clusters, such as Amazon Elastic Compute Cloud (Amazon EC2). See "CDH Whirr Installation" at: https://wiki.cloudera.com/display/DOC/Whirr+Installation
- Get help from the Cloudera Support team. Cloudera can help you install, configure, optimize, tune, and run Hadoop for large scale data processing and analysis. Cloudera supports Hadoop whether you run our distribution on servers in your own data center, or on hosted infrastructure services such as Amazon EC2, Rackspace, SoftLayer, or VMware's vCloud. For more information, see: http://www.cloudera.com/hadoop-support
- Get help from the community. Send a message to the CDH user's list: cdh-user@cloudera.org