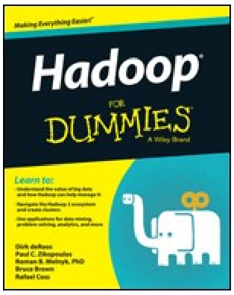

# Hadoop for Dummies

by Dirk deRoos et al.

# Chapter 3: Setting up Your Hadoop Environment

## In This Chapter

- Deciding on a Hadoop distribution
- Checking out the *Hadoop For Dummies* environment
- Creating your first Hadoop program: Hello Hadoop!

This chapter is an overview of the steps involved in actually getting started with Hadoop. We start with some of the things you need to consider when deciding which Hadoop distribution to use. It turns out that you have quite a few distributions to choose from, and any of them will make it easier for you to set up your Hadoop environment than if you were to go it alone, assembling the various components that make up the Hadoop ecosystem and then getting them to "play nice with one another." Nevertheless, the various distributions that are available *do* differ in the features that they offer, and the trick is to figure out which one is best for you.

This chapter also introduces you to the *Hadoop For Dummies* environment that we used to create and test all examples in this book. (If you're curious, we based our environment on Apache Bigtop.)

We round out this chapter with information you can use to create your first MapReduce program, after your Hadoop cluster is installed and running.

## Choosing a Hadoop Distribution

Commercial Hadoop distributions offer various combinations of open source components from the Apache Software Foundation and elsewhere — the idea is that the various components have been integrated into a single product, saving you the effort of having to assemble your own set of integrated components. In addition to open source software, vendors typically offer proprietary software, support, consulting services, and training.

How do you go about choosing a Hadoop distribution from the numerous options that are available? We provide an overview in Chapter 1 of the more prominent distributions, but when it comes to setting up your own environment, you're the one who has to choose, and that choice should be based on a set of criteria designed to help you make the best decision possible.

**REMEMBER** Not all Hadoop distributions have the same components (although they all have Hadoop's core capabilities), and not all components in one particular distribution are compatible with other distributions.

The criteria for selecting the most appropriate distribution can be articulated as this set of important questions:

- What do you want to achieve with Hadoop?
- How can you use Hadoop to gain business insight?
- What business problems do you want to solve?
- What data will be analyzed?
- Are you willing to use proprietary components, or do you prefer open source offerings?
- Is the Hadoop infrastructure that you're considering flexible enough for all your use cases?
- What existing tools will you want to integrate with Hadoop?
- Do your administrators need management tools? (Hadoop's core distribution doesn't include administrative tools.)
- Will the offering you choose allow you to move to a different product without obstacles such as vendor lock-in? (Application code that's not transferrable to other distributions or data stored in proprietary formats represent good examples of lock-in.)
- Will the distribution you're considering meet your future needs, insofar as you're able to anticipate those needs?

One approach to comparing distributions is to create a *feature matrix* — a table that details the specifications and features of each distribution you're considering. Your choice can then depend on the set of features and specs that best addresses the requirements around your specific business problems.

On the other hand, if your requirements include prototyping and experimentation, choosing the latest official Apache Hadoop distribution might prove to be the best approach. The most recent releases certainly have the newest most exciting features, but if you want stability you don't want excitement. For stability, look for an older release branch that's been available long enough to have some incremental releases (these typically include bug fixes and minor features).

Whenever you think about open source Hadoop distributions, give a moment's thought (or perhaps many moments' thought) to the concept of *open source fidelity* — the degree to which a particular distribution is compatible with the open source components on which it depends. High fidelity facilitates integration with other products that are designed to be compatible with those open source components. Low fidelity? Not so much.

**REMEMBER** The open source approach to software development itself is an important part of your Hadoop plans because it promotes compatibility with a host of third-party tools that you can leverage in your own Hadoop deployment. The open source approach also enables engagement with the Apache Hadoop community, which gives you, in turn, the opportunity to tap into a deeper pool of skills and innovation to enrich your Hadoop experience.

Because Hadoop is a fast-growing ecosystem, some parts continue to mature as the community develops tooling to meet industry demands. One aspect of this evolution is known as *backporting*, where you apply a new software modification or patch to a version of the software that's older than the version to which the patch is applicable. An example is NameNode failover: This capability is a part of Hadoop 2 but was backported (in its beta form) by a number of distributions into their Hadoop-1-based offerings for as much as a year before Hadoop 2 became generally available.

**TIP** Not every distribution engages actively in backporting new content to the same degree, although most do it for items such as bug fixes. If you want a production license for bleeding-edge technology, this is certainly an option; for stability, however, it's not a good idea.

The majority of Hadoop distributions include proprietary code of some kind, which frequently comes in the form of installers and a set of management tools. These distributions usually emerge from different business models. For example, one business model can be summarized this way: "Establish yourself as an open source leader and pioneer, market your company as having the best expertise, and sell that expertise as a service." Red Hat, Inc. is an example of a vendor that uses this model. In contrast to this approach, the embrace-and-extend business model has vendors building capabilities that extend the capabilities of open source software. MapR and IBM, which both offer alternative file systems to the Hadoop Distributed File System (HDFS), are good examples.

**TECHNICAL STUFF** People sometimes mistakenly throw the "fork" label at these innovations, making use of jargon used by software programmers to describe situations where someone takes a copy of an open source program as the starting point for their own (independent) development. The alternative file systems offered by MapR and IBM are completely different file systems, not a fork of the open source HDFS. Both companies enable their customers to choose either their proprietary distributed file system or HDFS. Nevertheless, in this approach, compatibility is critical, and the vendor must stay up to date with evolving interfaces. Customers need to know that vendors can be relied on to support their extensions.

## Choosing a Hadoop Cluster Architecture

Hadoop is designed to be deployed on a large cluster of networked computers, featuring master nodes (which host the services that control Hadoop's storage and processing) and slave nodes (where the data is stored and processed). You can, however, run Hadoop on a single computer, which is a great way to learn the basics of Hadoop by experimenting in a controlled space.

Hadoop has two deployment modes: pseudo-distributed mode and fully distributed mode, both of which are described below.

### Pseudo-Distributed Mode (Single Node)

A single-node Hadoop deployment is referred to as running Hadoop in *pseudo-distributed* mode, where all the Hadoop services, including the master and slave services, all run on a single compute node. This kind of deployment is useful for quickly testing applications while you're developing them without having to worry about using Hadoop cluster resources someone else might need. It's also a convenient way to experiment with Hadoop, as most of us don't have clusters of computers at our disposal. With this in mind, the *Hadoop for Dummies* environment is designed to work in pseudo-distributed mode.

### Fully Distributed Mode (A Cluster of Nodes)

A Hadoop deployment where the Hadoop master and slave services run on a cluster of computers is running in what's known as *fully distributed mode*. This is an appropriate mode for production clusters and development clusters. A further distinction can be made here: a *development cluster* usually has a small number of nodes and is used to prototype the workloads that will eventually run on a *production cluster*.

Chapter 16 provides extensive guidance on the hardware requirements for fully distributed Hadoop clusters with special considerations for both master and slave nodes as they have different requirements.

## The Hadoop for Dummies Environment

To help you get started with Hadoop, we're providing instructions on how to quickly download and set up Hadoop on your own laptop computer. As we mention earlier in the chapter, your cluster will be running in pseudo-distributed mode on a virtual machine, so you won't need special hardware.

A *virtual machine* (VM) is a simulated computer that you can run on a real computer. For example, you can run a program on your laptop that "plays" a VM, which opens a window that looks like it's running another computer. In effect, a pretend computer is running inside your real computer.

We'll be downloading a VM, and while running it, we'll install Hadoop.

**TIP** As you make your way through this book, enhance your learning by trying the examples and experimenting on your own!

### The Hadoop for Dummies Distribution: Apache Bigtop

We've done our best to provide a vendor-agnostic view of Hadoop with this book. It's with this in mind that we built the *Hadoop For Dummies* environment using Apache Bigtop, a great alternative if you want to assemble your own Hadoop components. Bigtop gathers the core Hadoop components for you and ensures that your configuration works. Apache Bigtop is a 100 percent open source distribution.

The primary goal of Bigtop — itself an Apache project, just like Hadoop — is to build a community around the packaging, deployment, and integration of projects in the Apache Hadoop ecosystem. The focus is on the system as a whole rather than on individual projects.

Using Bigtop, you can easily install and deploy Hadoop components without having to track them down in a specific distribution and match them with a specific Hadoop version. As new versions of Hadoop components are released, they sometimes do not work with the newest releases of other projects. If you're on your own, significant testing is required. With Bigtop (or a commercial Hadoop release) you can trust that Hadoop experts have done this testing for you. To give you an idea of how expansive Bigtop has gotten, see the following list of all the components included in Bigtop:

- Apache Crunch
- Apache Flume
- Apache Giraph
- Apache HBase
- Apache HCatalog
- Apache Hive
- Apache Mahout
- Apache Oozie
- Apache Pig
- Apache Solr
- Apache Sqoop
- Apache Whirr
- Apache Zookeeper
- Cloudera Hue
- LinkedIn DataFu

This collection of Hadoop ecosystem projects is about as expansive as it gets, as both major and minor projects are included. See Chapter 1 for summary descriptions of the more prominent projects.

Apache Bigtop is continuously evolving, so the list that's presented here was current at the time of writing. For the latest release information about Bigtop, visit http://blogs.apache.org/bigtop.

## Setting up the Hadoop for Dummies Environment

This section describes all the steps involved in creating your own *Hadoop For Dummies* working environment. If you're comfortable working with VMs and Linux, feel free to install Bigtop on a different VM than what we recommend. If you're really bold and have the hardware, go ahead and try installing Bigtop on a cluster of machines in fully distributed mode!

### Step 1: Downloading a VM

Hadoop runs on all popular Linux distributions, so we need a Linux VM. There is a freely available (and legal!) CentOS 6 image available here:

http://sourceforge.net/projects/centos-6-vmware

**WARNING!** You will need a 64-bit operating system on your laptop in order to run this VM. Hadoop needs a 64-bit environment.

After you've downloaded the VM, extract it from the downloaded Zip file into the destination directory. Do ensure you have around 50GB of space available as Hadoop and your sample data will need it.

If you don't already have a VM player, you can download one for free from here:

https://www.vmware.com/go/downloadplayer

After you have your VM player set up, open the player, go to File⇨Open, then go to the directory where you extracted your Linux VM. Look for a file called `centos-6.2-x64-virtual-machine-org.vmx` and select it. You'll see information on how many processors and how much memory it will use. Find out how much memory your computer has, and allocate half of it for the VM to use. Hadoop needs lots of memory.

Once you're ready, click the Play button, and your Linux instance will start up. You'll see lots of messages fly by as Linux is booting and you'll come to a login screen. The user name is already set to "Tom." Specify the password as "tomtom" and log in.

**Step 2: Downloading Bigtop**

From within your Linux VM, right-click on the screen and select Open in Terminal from the contextual menu that appears. This opens a Linux terminal, where you can run commands. Click inside the terminal so you can see the cursor blinking and enter the following command:

```
su -
```

You'll be asked for your password, so type "tomtom" like you did earlier. This command switches the user to root, which is the master account for a Linux computer — we'll need this in order to install Hadoop.

With your root access (don't let the power get to your head), run the following command:

```
wget -O /etc/yum.repos.d/bigtop.repo \]
      http://www.apache.org/dist/bigtop/bigtop-0.7.0/repos/centos6/bigtop.repo]
```

The `wget` command is essentially a web request, which requests a specific file in the URL we can see and writes it to a specific path — in our case, that's `/etc/yum.repos.d/bigtop.repo`.

**Step 3: Installing Bigtop**

The geniuses behind Linux have made life quite easy for people like us who need to install big software packages like Hadoop. What we downloaded in the last step wasn't the entire Bigtop package and all its dependencies. It was just a *repository file* (with the extension `.repo`), which tells an installer program which software packages are needed for the Bigtop installation.

Like any big software product, Hadoop has lots of prerequisites, but you don't need to worry. A well-designed `.repo` file will point to any dependencies, and the installer is smart enough to see if they're missing on your computer and then download and install them.

The installer we're using is called yum, which you get to see in action now:

```
yum install hadoop\* mahout\* oozie\* hbase\* hive\* hue\* pig\* zookeeper\*
```

Notice that we're picking and choosing the Hadoop components to install. There are a number of other components available in Bigtop, but these are the only ones we'll be using in this book. Since the VM we're using is a fresh Linux install, we'll need many dependencies, so you'll need to wait a bit. The yum installer is quite verbose, so you can watch exactly what's being downloaded and installed to pass the time. When the install process is done, you should see a message that says "Complete!"

**Step 4: Starting Hadoop**

Before we start running applications on Hadoop, there are a few basic configuration and setup things we need to do. Here they are in order:

1. Download and install Java:

   ```
   yum install java-1.7.0-openjdk-devel.x86_64
   ```

2. Format the NameNode:

   ```
   sudo /etc/init.d/hadoop-hdfs-namenode init
   ```

3. Start the Hadoop services for your pseudodistributed cluster:

   ```
   for i in hadoop-hdfs-namenode hadoop-hdfs-datanode ; \
       do sudo service $i start ; done
   ```

4. Create a sub-directory structure in HDFS:

   ```
   sudo /usr/lib/hadoop/libexec/init-hdfs.sh
   ```

5. Start the YARN daemons:

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

And with that, you're done. Congratulations! You've installed a working Hadoop deployment!

## The Hadoop for Dummies Sample Data Set: Airline on-Time Performance

Throughout this book, we'll be running examples based on the Airline On-time Performance data set — we call it the flight data set for short. This data set is a collection of all the logs of domestic flights from the period of October 1987 to April 2008. Each record represents an individual flight, where various details are captured, such as the time and date of arrival and departure, the originating and destination airports, and the amount of time taken to taxi from the runway to the gate. For more information about this data set see this page: http://stat-computing.org/dataexpo/2009/.

Many of us on the author team for this book spend a lot of time on planes, so this example data set is close to our hearts.

**Step 5: Downloading the Sample Data Set**

To download the sample data set, open the Firefox browser from within the VM, and go to the following page: http://stat-computing.org/dataexpo/2009/the-data.html.

You won't need the entire data set, so we recommend you start with a single year, so select 1987. When you're about to download, select the Open with Archive Manager option.

After your file has downloaded, extract the `1987.csv` file into your home directory where you'll easily be able to find it. Click on the Extract button, and then select the Desktop directory.

### Step 6: Copying the Sample Data Set into HDFS

Remember that your Hadoop programs can only work with data once it's stored in HDFS. So what we're going to do now is copy the flight data file for 1987 into HDFS. Enter the following command:

```
hdfs dfs -copyFromLocal 1987.csv /user/root
```

## Your First Hadoop Program: Hello Hadoop!

After the Hadoop cluster is installed and running, you can run your first Hadoop program.

This application is very simple, and calculates the total miles flown for all flights flown in one year. The year is defined by the data file you read in your application. We will look at MapReduce programs in more detail in Chapter 6, but to keep things a bit simpler here, we'll run a Pig script to calculate the total miles flown. You will see the map and reduce phases fly by in the output.

Here is the code for this Pig script:

```
records = LOAD '2013_subset.csv' USING PigStorage(',') AS
              (Year,Month,DayofMonth,DayOfWeek,DepTime,CRSDepTime,ArrTime,\

              CRSArrTime,UniqueCarrier,FlightNum,TailNum,ActualElapsedTime,\
              CRSElapsedTime,AirTime,ArrDelay,DepDelay,Origin,Dest,\
              Distance:int,TaxiIn,TaxiOut,Cancelled,CancellationCode,\
              Diverted,CarrierDelay,WeatherDelay,NASDelay,SecurityDelay,\
              LateAircraftDelay);
milage_recs = GROUP records ALL;
tot_miles = FOREACH milage_recs GENERATE SUM(records.Distance);
STORE tot_miles INTO /user/root/totalmiles;
```

We want to put this code in a file on our VM, so let's first create a file. Right-click on the desktop of your VM and select Create Document from the contextual menu that appears and name the document `totalmiles.pig`. Then open the document in an editor, paste in the above code, and save the file.

From the command line, run the following command to run the Pig script:

```
pig totalmiles.pig
```

You will see many lines of output, and then finally a "Success!" message, followed by more statistics, and then finally the command prompt. After your Pig job has completed, you can see your output:

```
hdfs dfs -cat /user/root/totalmiles/part-r-00000
```

Drumroll, please… And the answer is:

```
775009272
```

And with that, you've run your first Hadoop application! The examples in this book use the flight data set, and will work in this environment, so do be sure to try them out yourself.