# AWS Setup

## Setting up your AWS account

Note: Amazon will ask you for your credit card information during the setup process. This is normal.

1. Go to http://aws.amazon.com/ and sign up:
   a. You may sign in using your existing Amazon account or you can create a new account by selecting "I am a new user."
   b. Enter your contact information and confirm your acceptance of the AWS Customer Agreement.
   c. Once you have created an Amazon Web Services Account, you may need to accept a telephone call to verify your identity. Some students have used Google Voice successfully if you don't have or don't want to give a mobile number. You need Access Identifiers to make valid web service requests.
2. Go to http://aws.amazon.com/ and sign in. You need to double-check that your account is signed up for three of their services: Simple Storage Service (S3), Elastic Compute Cloud (EC2), and Amazon Elastic MapReduce by clicking here -- you should see "Services You're Signed Up For" under "Manage Your Account".
3. You should have received your AWS credit code by email or in class. Armed with this code, go to http://aws.amazon.com/awscredits/ This step will give you $100 credit towards AWS. Be aware that if you exceed it, amazon will charge your credit card without warning. Normally, this credit is more than enough for this homework assignment (if you are interested in their changes, see AWS charges: currently, AWS charges about 10 cents/node/hour for the default "small" node size.). However, **you must remember to terminate *manually* the AWS clusters when you are done**: if you just close the browser, the clusters continue to run, and amazon will continue to charge you for days and weeks, exhausting your credit and charging you huge amount on your credit card. Remember to terminate the AWS cluster.

## Setting up an EC2 key pair

To connect to an Amazon EC2 node, such as the master nodes for the Hadoop clusters you will be creating, you need an SSH key pair. To create and install one, do the following:

1. Go to AWS security credentials page and make sure that you see a key under Access Keys. If not just click Create a new Access Key.
2. Go to the EC2 Management Console. Click "Key Pairs" on the navigation panel. Then click the "Create Key Pair" button. Enter a key pair name and click "Yes". (Don't do this in Internet Explorer, or you might not be able to download the .pem private key file.)
3. Download and save the .pem private key file to disk. We will reference the .pem file as `</path/to/saved/keypair/file.pem>` in the following instructions.
4. Make sure only you can access the .pem file. If you do not change the permissions, you will get an error message later:

```
$ chmod 400 </path/to/saved/keypair/file.pem>
```

Note: This step will NOT work on Windows 7 with cygwin. Windows 7 does not allow file permissions to be changed through this mechanism, and they must be changed for ssh to work. So if you must use Windows, you should use PuTTY as your ssh client. In this case, you will further have to transform this key file into PuTTY format. For more information go to Amazon's instruction on EC2 Instance connection using PuTTY and follow step 1 and 2. The rest of the steps can be followed to connect to EC2 instance once you start an AWS cluster in the next section.

## Starting an MapReduce Cluster and running Pig Interactively

To run a Pig job on AWS, you need to start up an cluster using the Elastic MapReduce Management Console and connect to the Hadoop master node. Follow the steps below. You may also find Amazon's interactive Pig tutorial

useful, but note that the screenshots are slightly out of date.

To set up and connect to a pig cluster, perform the following steps:

1. Go to http://console.aws.amazon.com/elasticmapreduce/vnext/home and sign in.
2. Click the "Create Cluster" button.
3. In the "Cluster name" field, type a name such as "Pig Interactive Job Flow" or "Homework 8"
4. Uncheck "Enabled" for "Logging"
5. In the "Hardware Configuration" section, change the count of core instances to 1. (In the homework problems you will need more core instances, rather than just 1.)
6. In the "Security and Access" section, select the EC2 key pair you created above. **Make sure to select a key pair**, otherwise you won't be able to ssh to the cluster and run jobs.
7. In the "Bootstrap Actions" section, select "Memory intensive configuration" from the dropdown list. Click "Configure and add", and then click "Add". You need to do this step because the default configuration can sometimes run into memory problems.
8. Click "Create cluster" at the bottom of the page. You can go back to the cluster list and should see the cluster you just created. It may take a few minutes for the cluster to launch. If your cluster fails or takes an extraordinarily long time, Amazon may be near capacity. Try again later. If it still doesn't work, contact the TA.

9. Now you need to obtain the Master Public DNS. Click on cluster name. You will find the Master Public DNS at the top. We call this Master Public DNS name **<master.public-dns-name.amazonaws.com>**.

10. When the status becomes Running, you can connect to your cluster and run Pig jobs. From a terminal, use the following command:

```
$ ssh -o "ServerAliveInterval 10" -i </path/to/saved/keypair/file.pem>
 hadoop@<master.public-dns-name.amazonaws.com>
```

11. Once you connect successfully, just type

```
$ pig
```

12. Now you should have a Pig prompt:

```
grunt>
```

*Note*: When you first connect to the cluster and type `pig` in command line, it might say "command not found". Wait a few minutes and try again. The Pig environment takes a while to set up.
This is the interactive mode where you type in pig queries. You are now ready to return to the homework assignment. In this homework we will use pig only interactively. (The alternative is to have pig read the program from a file.)

Other useful information:

- For the first job you run, Hadoop will create the output directory for you automatically. But Hadoop refuses to overwrite existing results. So you will need to move your prior results to a different directory before re-running your script, specify a different output directory in the script, or delete the prior results altogether.
  To see how to perform these tasks and more, see "Managing the results of your Pig queries" below.
- To exit pig, type `quit` at the `grunt>` promt. To terminate the ssh session, type `exit` at the unix prompt: after that you must terminate the AWS cluster (see next).
- To kill a pig job type CTRL/C while pig is running.This kills pig only: after that you need to kill the hadoop job. We show you how to do this below.

# Monitoring Hadoop jobs

You are required in this homework to monitor the running Hadoop jobs on your AWS cluster using the master node's *job tracker* web UI.

By far the easiest way to do this is to use ssh tunneling.

1. Add *listening* options to your `ssh` command

```
ssh -L 9100:localhost:9100 -L 9101:localhost:9101 -o "ServerAliveInterval 10" -i
</path/to/saved/keypair/file.pem> hadoop@<master.public-dns-name.amazonaws.com>
```

2. Open your browser to http://localhost:9100

From there, you can monitor your jobs' progress using the UI.

There are two other ways to do this: using lynx or using your own browser with a SOCKS proxy.

1. Using LYNX. Very easy, you don't need to download anything. Open a separate `ssh` connection to the AWS master node and type:

```
$ lynx http://localhost:9100/
```

   Lynx is a text browswer. Navigate as follows: `up/down arrows` = move through the links (current link is highlighted); `enter` = follows a link; `left arrow` = return to previous page.

   Examine the webpage carefully, while your pig pgram is running. You should find information about the map tasks, the reduce tasks, you should be able to drill down into each map task (for example to monitor its progress); you should be able to look at the log files of the map tasks (if there are runtime errors, you will see them only in these log files).

2. Using SOCKS proxy, and your own browser. This requires more work, but the nicer interface makes it worth the extra work
   1. Set up your browser to use a proxy when connecting to the master node. *Note: If the instructions fail for one browser, try the other browser*. In particular, it seems like people are having problems with Chrome but Firefox, especially following Amazon's instructions, works well.
      - Firefox:
        1. Install the FoxyProxy extension for Firefox.li>
        2. Copy the `foxyproxy.xml` configuration file from the `hw8/` folder into your Firefox profile folder.
        3. If the previous step doesn't work for you, try deleting the `foxyproxy.xml` you copied into your profile, and using Amazon's instructions to set up FoxyProxy manually. If you use Amazon's instructions, be careful to use port 8888 instead of the port in the instructions.
      - Chrome:
        1. Option 1: FoxyProxy is now available for Chrome as well.
        2. Option 2: You can try proxy switch!
        3. Click the *Tools* icon (upper right corner; don't confuse it with the Developer's Tools !), Go to *Tools*, go to *Extensions*. Here you will see the ProxySwitch!: click on *Options*.
        4. Create a new Proxy Profile: Manual Configuration, Profile name = Amazon Elastic MapReduce (any name you want), SOCKS Host = localhost, Port = 8888 (you can choose any port you want; another favorite is 8157), SOCKS v5. If you don't see "SOCKS", de-select the option to "Use the same proxy server for all protocols".
        5. Create two new switch rules (give them any names, say AWS1 and AWS2). Rule 1: pattern=*.amazonaws.com:*/*, Rule 2: pattern=*.ec2.internal:*/*. For both, Type=wildcard, Proxy profile=[the profile you created at the previous step].
   2. Open a new local terminal window and create the SSH SOCKS tunnel to the master node using the following:

```
$ ssh -o "ServerAliveInterval 10" -i </path/to/saved/keypair/file.pem> -ND 8888
hadoop@<master.public-dns-name.amazonaws.com>
```

      (The `-N` option tells `ssh` not to start a shell, and the `-D 8888` option tells `ssh` to start the proxy and have it listen on port 8888.)

The resulting SSH window will appear to hang, without any output; this is normal as SSH has not started a shell on the master node, but just created the tunnel over which proxied traffic will run.

Keep this window running in the background (minimize it) until you are finished with the proxy, then close the window to shut the proxy down.

3. Open your browser, and type one of the following URLs:
   - For the job tracker: `http://<master.public-dns-name.amazonaws.com>:9100/`
   - For HDFS management: `http://<master.public-dns-name.amazonaws.com>:9101/`

The job tracker enables you to see what MapReduce jobs are executing in your cluster and the details on the number of maps and reduces that are running or already completed.

Note that, at this point in the instructions, you will not see any MapReduce jobs running but you should see that your cluster has the capacity to run a couple of maps and reducers on your one instance.

The HDFS manager gives you more low-level details about your cluster and all the log files for your jobs.

# Killing a Hadoop Job

Later, in the assignment, we will show you how to launch MapReduce jobs through Pig. You will basically write Pig Latin scripts that will be translated into MapReduce jobs (see lecture notes). Some of these jobs can take a long time to run. If you decide that you need to interrupt a job before it completes, here is the way to do it:

If you want to kill pig, you first type CTRL/C, which kills pig only. Next, kill the hadoop job, as follows. From the job tracker interface find the hadoop `job_id`, then type:

```
$ hadoop job -kill job_id
```

# Terminating an AWS cluster

When you are done running Pig scripts, make sure to **ALSO** terminate your cluster. This is a step that you need to do **in addition to** stopping pig and Hadoop (if necessary) above.

This step shuts down your AWS cluster:

1. Go to the MapReduce Management Console.
2. Select the cluster in the list.
3. Click the Terminate button. Turn off Termination protection so you can terminate the cluster.
4. Wait for a while (may take minutes) and recheck until the job state becomes TERMINATED.

**Pay attention to this step**. If you fail to terminate your job and only close the browser, or log off AWS, your AWS will continue to run, and AWS will continue to charge you: for hours, days, weeks, and when your credit is exhausted, it chages your creditcard. Make sure you don't leave the console until you have confirmation that the job is terminated.

You can now shut down your cluster.

# Checking your Balance

Please check your balance regularly!!!

1. Go to your Account Homepage.
2. Click "Account Activity" on the navigation panel.
3. Now click on "detail" to see any charges < $1.

To avoid unnecessary charges, terminate your clusters when you are not using them.

**USEFUL**: AWS customers can now use **billing alerts** to help monitor the charges on their AWS bill. You can visit your Billing Console to enable monitoring of your charges. Then, you can set up a billing alert by simply specifying a bill threshold and an e-mail address to be notified as soon as your estimated charges reach the threshold.

# Managing the results of your Pig queries

For the next step, you need to restart a new cluster as follows. Hopefully, it should now go very quickly:

- Start a new cluster with one instance.
- Start a new interactive Pig session (through grunt)
- Start a new SSH SOCKS tunnel to the master node (if you are using your own browser)

We will now get into more details about running Pig scripts.

Your pig program stores the results in several files in a directory. You have two options: (1) store these files in the Hadoop File System, or (2) store these files in S3. In both cases you need to copy them to your local machine.

## 1. Storing Files in the Hadoop File System

This is done through the following pig command (used in `example.pig`):

```
store count_by_object_ordered into '/user/hadoop/example-results' using PigStorage();
```

Before you run the pig query, you need to (A) create the /user/hadoop directory. After you run the query you need to (B) copy this directory to the local directory of the AWS master node, then (C) copy this directory from the AWS master node to your local machine.

### 1.A. Create the "/user/hadoop Directory" in the Hadoop Filesystem

You will need to do this for each new cluster that you create.

To create a `/user/hadoop` directory on the AWS cluster's HDFS file system run this from the AWS master node:

```
$ hadoop dfs -mkdir /user/hadoop
```

Check that the directory was created by listing it with this command:

```
$ hadoop dfs -ls /user/hadoop
```

You may see some output from either command, but you should not see any errors.

You can also do this directly from grunt with the following command.

```
grunt> fs -mkdir /user/hadoop
```

Now you are ready to run your first sample program. Take a look at the starter code that we provided in hw8.tar.gz. Copy and paste the content of `example.pig`. (We give more details about this program back in

hw8.html).

**Note**: The program may appear to hang with a 0% completion time... go check the job tracker. Scroll down. You should see a MapReduce job running with some non-zero progress.

**Note 2**: Once the first MapReduce job gets to 100%... if your grunt terminal still appears to be suspended... go back to the job tracker and make sure that **the reduce phase is also 100% complete**. It can take some time for the reducers to start making any progress.

**Note 3**: The example generates more than 1 MapReduce job... so be patient.

### 1.B. Copying files from the Hadoop Filesystem

The result of a pig script is stored in the hadoop directory specified by the `store` command. That is, for `example.pig`, the output will be stored at `/user/hadoop/example-results`, as specified in the script. HDFS is separate from the master node's file system, so before you can copy this to your local machine, you must copy the directory from HDFS to the master node's Linux file system:

```
$ hadoop dfs -copyToLocal /user/hadoop/example-results example-results
```

This will create a directory `example-results` with `part-*` files in it, which you can copy to your local machine with `scp`. You can then concatenate all the `part-*` files to get a single results file, perhaps sorting the results if you like.

An easier option may be to use

```
$ hadoop fs -getmerge  /user/hadoop/example-results example-results
```

This command takes a source directory and a destination file as input and concatenates files in src into the destination local file.

Use `hadoop dfs -help` or see the `hadoop dfs` guide to learn how to manipulate HDFS. (Note that `hadoop fs` is the same as `hadoop dfs`.)

### 1.C. Copying files to or from the AWS master node

- To copy one file from the master node back to your computer, run this command *on the local computer:*

  ```
  $ scp -o "ServerAliveInterval 10" -i </path/to/saved/keypair/file.pem>
  hadoop@<master.public-dns-name.amazonaws.com>:<file_path> .
  ```

  where `<file_path>` can be absolute or relative to the AWS master node's home folder. The file should be copied onto your current directory ('.') on your local computer.

- Better: copy an entire directory, recursively. Suppose your files are in the directory `example-results`. They type the following *on your loal computer*:

  ```
  $ scp -o "ServerAliveInterval 10" -i </path/to/saved/keypair/file.pem> -r
  hadoop@<master.public-dns-name.amazonaws.com>:example-results .
  ```

- As an alternative, you may run the scp command on the AWS master node, and connect to your local machine. For that, you need to know your local machine's domain name, or IP address, and your local machine needs to accept ssh connections.

## 2. Storing Files in S3

To use this approach, go to your AWS Management Console, click on Create Bucket, and create a new bucket (=directory). Give it a name that may be a public name. Do not use any special chatacters, including underscore. Let's say you call it `supermanhw8`. Click on Actions, Properties, Permissions. Make sure you have all the permissions.

Modify the store command of `example.pig` to:

```
store count_by_object_ordered into 's3n://supermanhw8/example-results';
```

Run your pig program. When it terminates, then in your S3 console you should see the new directory `example-results`. Click on individual files to download. The number of files depends on the number of reduce tasks, and may vary from one to a few dozens. The only disadvantage of using S3 is that you have to click on each file separately to download.

Note that S3 is permanent storage, and you are charged for it. You can safely store all your query answers for several weeks without exceeding your credit; at some point in the future remember to delete them.