

# Spinning up an Apache Spark Cluster: Step-by-Step

05 OCTOBER 2015

Austin Ouyang is an *Insight Data Labs* lead mentor and Data Engineer & Program Director at Insight.

The DevOps series covers how to get started with the leading open source distributed technologies. In this tutorial, we step through how to deploy a Spark Standalone cluster on AWS for less than \$1. In a follow up post, we will show you how to use a Jupyter notebook on Spark for ad hoc analysis of Reddit comment data on Amazon S3.

One of the significant hurdles in learning to build distributed systems is understanding how these various technologies are installed and their inter-dependencies. In our experience, the best way to get started with these technologies is to roll up your sleeves and build projects you are passionate about.

This following tutorial shows how you can deploy your own Spark cluster in standalone mode on top of Hadoop. Due to Spark's memory demand, we recommend using m4.large spot instances with 200GB of magnetic hard drive space each.

m4.large spot instances are not within the free-tier package on AWS, so this tutorial will incur a small cost. The tutorial should not take any longer than a couple hours, but if we allot 6 hours for your 4 node spot cluster, the total cost should run around \$0.69 depending on the region of your cluster. If you run this cluster for an entire month we can look at a bill of around \$80, so be sure to spin down you cluster after you are finished using it.

## Spark Introduction

### Spinning up AWS EC2 Spot Instances

### Install Hadoop

### Install Spark

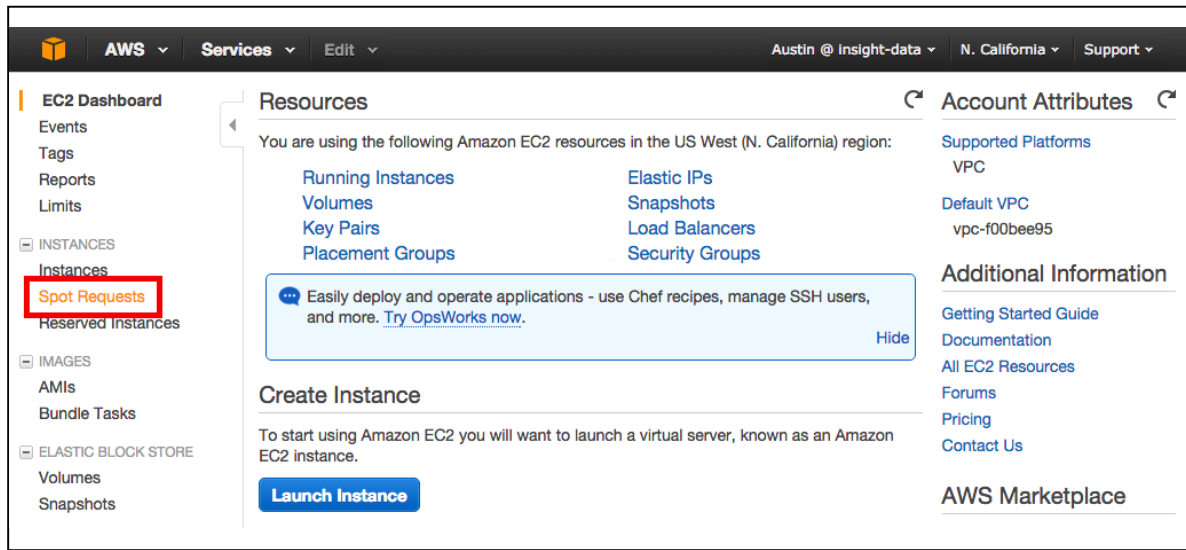
### Start Your Spark Cluster

# Spark Introduction

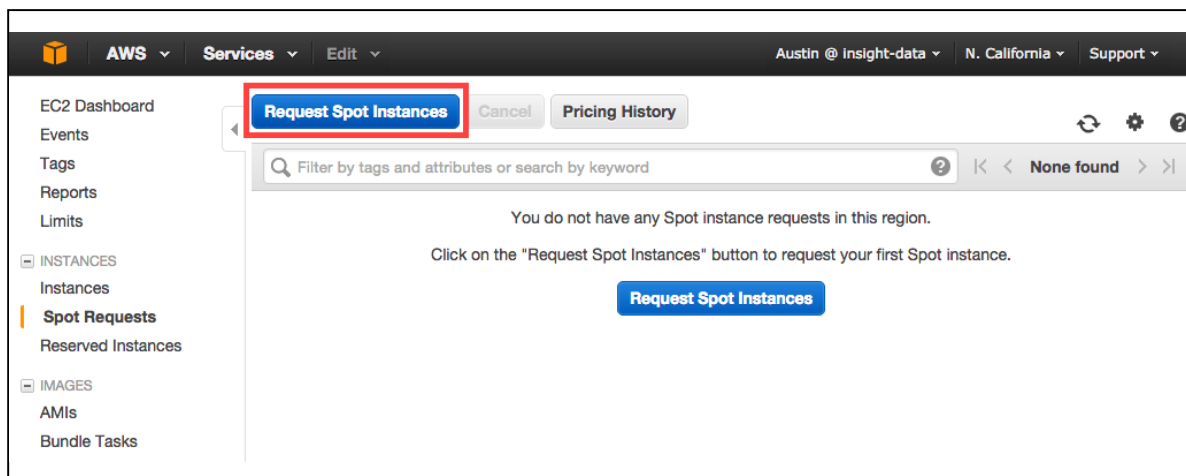
Spark is a general computation engine that uses distributed memory to perform fault-tolerant computations with a cluster. Even though Spark is relatively new, it is one of the hottest open-source technologies at the moment and has begun to surpass Hadoop's MapReduce model. This is partly because Spark's Resilient Distributed Dataset (RDD) model can do everything the MapReduce paradigm can, and more. In addition, Spark can perform iterative computations at scale, which opens up the possibility of executing machine learning algorithms (with Spark MLlib) much faster than with Hadoop alone.

# Spinning up AWS EC2 Spot Instances

Spot instances are great for reducing AWS costs sometimes cutting instance costs by a whole order of magnitude. The tradeoff is that you can lose the instances at any time if the demand for a specific instance type increases past your bid price. Spinning a spot instance for this tutorial is just as easy as on-demand instances. In your AWS EC2 dashboard, you will need to click on Spot Requests instead of Instances.



Next request the spot instances.



We will be using the Ubuntu 14.04 AMI.

**Step 1: Choose an Amazon Machine Image (AMI)**

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

**Quick Start**

My AMIs  
AWS Marketplace  
Community AMIs  
**Free tier only**

**Amazon Linux AMI 2015.03 (HVM), SSD Volume Type** - ami-d114f295  
The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.  
Root device type: ebs Virtualization type: hvm  
**Select** 64-bit

**Red Hat Enterprise Linux 7.1 (HVM), SSD Volume Type** - ami-a540a5e1  
Red Hat Enterprise Linux version 7.1 (HVM), EBS General Purpose (SSD) Volume Type  
Root device type: ebs Virtualization type: hvm  
**Select** 64-bit

**SUSE Linux Enterprise Server 12 (HVM), SSD Volume Type** - ami-b95b4ffc  
SUSE Linux Enterprise Server 12 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.  
Root device type: ebs Virtualization type: hvm  
**Select** 64-bit

**Ubuntu Server 14.04 LTS (HVM), SSD Volume Type** - ami-df6a8b9b  
Ubuntu Server 14.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).  
**Select** 64-bit

Next select the type of instance you would like to spin up. The minimum is m4.large which is what we will use for this tutorial.

**Step 2: Choose an Instance Type**

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All instance types Current generation Show/Hide Columns

Currently selected: m4.large (6.5 ECUs, 2 vCPUs, 2.4 GHz, Intel Xeon E5-2676v3, 8 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
<input type="radio"/>	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate
<input type="radio"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate
<input type="radio"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate
<input type="radio"/>	General purpose	t2.large	2	8	EBS only	-	Low to Moderate
<input checked="" type="checkbox"/>	General purpose	m4.large	2	8	EBS only	Yes	Moderate
<input type="checkbox"/>	General purpose	m4.xlarge	4	16	EBS only	Yes	High
<input type="checkbox"/>	General purpose	m4.2xlarge	8	32	EBS only	Yes	High
<input type="checkbox"/>	General purpose	m4.4xlarge	16	64	EBS only	Yes	High

[Cancel](#) [Previous](#) [Next: Configure Instance Details](#)

Moving on to the spot instance details, set the number of instances to 4 and the maximum price to \$0.02. In general if you place a maximum price about 50% higher

than the current price, you should not have instances terminate that often.

**Step 3: Configure Instance Details**  
Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

**Number of instances** ⓘ 4

**Purchasing option** ⓘ ☒ Request Spot instances

**Current price** ⓘ  
us-west-1a 0.0143  
us-west-1c 0.0147

**Maximum price** ⓘ \$ 0.02

**Launch group** ⓘ (Optional)

**Request valid from** ⓘ Any time [Edit](#)

**Request valid to** ⓘ Any time [Edit](#)

**Persistent request** ⓘ ☐ Persistent request

**Network** ⓘ vpc-f00bee95 (172.31.0.0/16) (default) [Create new VPC](#)

**Subnet** ⓘ No preference (default subnet in any Availability Zone) [Create new subnet](#)

**Auto-assign Public IP** ⓘ Use subnet setting (Enable)

**Placement group** ⓘ No placement group

**IAM role** ⓘ None [Create new IAM role](#)

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Add Storage](#)

For this tutorial we will also bump up the storage space per instance to 200GB. Choosing the magnetic option can help cut down costs, but have worse read and write performance than SSDs.

AWS

Services

Edit

Austin @ Insight-data

N. California

Support

1. Choose AMI

2. Choose Instance Type

3. Configure Instance

4. Add Storage

5. Tag Spot Request

6. Configure Security Group

7. Review

### Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more](#) about storage options in Amazon EC2.

Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Delete on Termination	Encrypted
Root	/dev/sda1	snap-ccea80f5	200	Magnetic	N/A	<input checked="" type="checkbox"/>	Not Encrypted

Add New Volume

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. [Learn more](#) about free usage tier eligibility and usage restrictions.

Cancel

Previous

Review and Launch

Next: Tag Spot Request

Next name your instances.

AWS

Services

Edit

Austin @ Insight-data

Oregon

Support

1. Choose AMI

2. Choose Instance Type

3. Configure Instance

4. Add Storage

5. Tag Spot Request

6. Configure Security Group

### Step 5: Tag Spot Request

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. [Learn more](#) about tagging your Amazon EC2 resources.

Note that these tags will be applied to this Spot instance request and not to any instances launched to fulfill this request.

Key (127 characters maximum)	Value (255 characters maximum)
Name	spark-cluster

Create Tag (Up to 10 tags maximum)

Cancel

Previous

Review and Launch

Next: Configure Security Group

The next step will configure the security groups setting for these spot instances. All the ports are open for simplicity, but it should be noted that these settings should be

much more strict if put in production. If a security group does not exist with the following configuration, you can create a new security group with the following settings.

**Step 6: Configure Security Group**

Assign a security group: ☒ Create a new security group  
☐ Select an existing security group

Security group name: **OPEN\_TO\_ALL**

Description: launch-wizard-19 created 2015-10-04T23:24:31.284-07:00

Type	Protocol	Port Range	Source
All traffic	All	0 - 65535	Anywhere 0.0.0.0/0

**Warning**  
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

[Cancel](#) [Previous](#) [Review and Launch](#)

You will then be asked to choose which option for your instances' boot volume. Choose the magnetic option.

**Boot from General Purpose (SSD)**

General Purpose (SSD) volumes provide the ability to burst to 3000 IOPS per volume, independent of volume size, to meet the performance needs of most applications and also deliver a consistent baseline of 3 IOPS/GiB.

- ☐ Make General Purpose (SSD) the default boot volume for all instance launches from the console going forward (recommended).
- ☐ Make General Purpose (SSD) the boot volume for this instance.
- ☒ Continue with Magnetic as the boot volume for this instance.

Free tier eligible customers can get up to 30GB of General Purpose (SSD) storage.

☐ Don't show again [Next](#)

The final step will let you review your configurations before launching. Be sure that the AMI, instance type, security groups, and number of instances are correct.

**Step 7: Review Spot Instance Request**

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Spot Request 6. Configure Security Group

**AMI Details** [Edit AMI](#)

**Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - ami-5189a661**  
Free tier eligible  
Ubuntu Server 14.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).  
Root Device Type: ebs Virtualization type: hvm

**Instance Type** [Edit instance type](#)

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
m4.large	6.5	2	8	EBS only	Yes	Moderate

**Security Groups** [Edit security groups](#)

Security group name: OPEN\_TO\_ALL  
Description: launch-wizard-19 created 2015-10-04T23:27:24.514-07:00

Type	Protocol	Port Range	Source
All traffic	All	All	0.0.0.0/0

**Instance Details** [Edit instance details](#)

Number of instances: 4  
Network: vpc-742adf11  
Purchasing option: Spot  
Maximum price: 0.02

[Cancel](#) [Previous](#) [Launch](#)

## Install Hadoop

Spark uses several dependencies from Hadoop. In case you have not installed Hadoop on your cluster, please do so from [our Hadoop DevOps post](#) before proceeding.

## Install Spark

We will be using our Hadoop NameNode to act as our Spark Master and the remaining Hadoop DataNodes as Spark Workers. We will first install Spark on all the nodes and then configure the Spark Master. We recommend having 4 terminals open with each terminal representing a node. If you are using iTerm2, toggling the broadcast input to all panes can help reduce mistakes during installation.



We assume Java has been previously installed. You can check for this with the following command

```
allnodes$ java -version

java version "1.7.0_79"

OpenJDK Runtime Environment (IcedTea 2.5.6) (7u79-2.5.6-0ubuntu1.14.04.1)

OpenJDK 64-Bit Server VM (build 24.79-b02, mixed mode)
```

Install Scala with the following:

```
allnodes$ sudo apt-get install scala
```

We can test to see if Scala installed correctly with the following command:

```
allnodes$ scala -version

Scala code runner version 2.9.2 -- Copyright 2002-2011, LAMP/EPFL
```

Next install Spark 1.4.1 onto all the nodes by first saving the binary tar files to `~/Downloads` and extracting it to the `/usr/local` folder:

```
allnodes$ wget http://apache.mirrors.tds.net/spark/spark-1.4.1/spark-1.4.1-bin-hadoop2.4.tgz -P ~/Downloads

allnodes$ sudo tar zxvf ~/Downloads/spark-* -C /usr/local

allnodes$ sudo mv /usr/local/spark-* /usr/local/spark
```

## Environment Variables

---

Now add the Spark environment variables to `~/.profile` and source it to the current shell session.

### **~/.profile**

```
export SPARK_HOME=/usr/local/spark  
export PATH=$PATH:$SPARK_HOME/bin
```

Then load these environment variables by sourcing the profile:

```
allnodes$ . ~/.profile
```

Change the ownership of the `$SPARK_HOME` directory to the user ubuntu:

```
allnodes$ sudo chown -R ubuntu $SPARK_HOME
```

## Spark Configurations

For a basic setup of Spark, we will create a Spark configuration file using the existing template that comes with Spark. All the current configuration changes will be applied to the Spark Master node and all the Spark Worker nodes. After these changes, we will apply configurations specific to the Spark Master node.

## Common Spark Configurations on all Nodes

The only file to focus on is the `$SPARK_HOME/conf/spark-env.sh`. First make a copy of the template and rename it:

```
allnodes$ cp $SPARK_HOME/conf/spark-env.sh.template
$SPARK_HOME/conf/spark-env.sh
```

The entire file should be commented out, so we can insert our configurations wherever we would like. For now insert them near the beginning as shown below:

### **`$SPARK_HOME/conf/spark-env.sh`**

```
#!/usr/bin/env bash

# This file is sourced when running various Spark programs.
# Copy it as spark-env.sh and edit that to configure Spark for your site.

export JAVA_HOME=/usr

export SPARK_PUBLIC_DNS="current_node_public_dns"

export SPARK_WORKER_CORES=6

...

...

...
```

We have chosen `SPARK_WORKER_CORES` to be 6 based on the instances that we are using for this example cluster setup. In general, this variable defines the amount of parallelism each Spark Worker node has. The variable `SPARK_WORKER_CORES` can be misleading, since it does not represent the number of physical cores on your Spark Worker machine. Instead it represents the number of Spark tasks (or threads) a Spark Worker can give to its Spark Executors.

Our Spark Worker nodes are m4.large instances containing 2 CPU cores. Assuming that only one Spark Executor is spawned by the Spark Worker, this means that at most 6 Spark tasks can be distributed amongst the 2 CPUs for one Spark Application at any given time. Here we are oversubscribing by a factor of 3. By default, if this

value is not set, there will be no oversubscription of Spark Worker Cores (Spark task slots). In most cases you should always oversubscribe by at least a factor of 2 and will typically see a significant performance benefit.

## Spark Master Specific Configurations

Now that all the common configurations are complete, we will finish up the Spark Master specific configurations. Create a slaves file in the Spark configuration folder which will contain the public DNS's of all the Spark Worker nodes:

```
spark_master_node$ touch $SPARK_HOME/conf/slaves
```

### **\$SPARK\_HOME/conf/slaves**

```
spark_worker1_public_dns  
spark_worker2_public_dns  
spark_worker3_public_dns
```

## Start Your Spark Cluster

We can now start up the Spark cluster from the Spark Master node

```
spark_master_node$ $SPARK_HOME/sbin/start-all.sh
```

You can go to `spark_master_public_dns:8080` in your browser to check if all Worker nodes are online. If the webUI does not display, check to make sure your EC2 instances have security group settings that include All Traffic and not just SSH.



## Spark Master at spark://ip-172-31-38-214:7077

URL: spark://ip-172-31-38-214:7077

REST URL: spark://ip-172-31-38-214:6066 (cluster mode)

Workers: 3

Cores: 18 Total, 0 Used

Memory: 8.6 GB Total, 0.0 B Used

Applications: 0 Running, 0 Completed

Drivers: 0 Running, 0 Completed

Status: ALIVE

### Workers

Worker Id	Address	State	Cores	Memory
<a href="#">worker-20150929091656-172.31.38.215-44824</a>	172.31.38.215:44824	ALIVE	6 (0 Used)	2.9 GB (0.0 B Used)
<a href="#">worker-20150929091656-172.31.38.216-47400</a>	172.31.38.216:47400	ALIVE	6 (0 Used)	2.9 GB (0.0 B Used)
<a href="#">worker-20150929091656-172.31.38.217-39731</a>	172.31.38.217:39731	ALIVE	6 (0 Used)	2.9 GB (0.0 B Used)

### Running Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
----------------	------	-------	-----------------	----------------	------	-------	----------

### Completed Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
----------------	------	-------	-----------------	----------------	------	-------	----------

## Next Steps

Now that you have a working Spark cluster you can start creating your own RDDs, performing operations on RDDs, and reading and writing to HDFS, S3, Cassandra or many other distributed file systems and databases. In our next tutorial, we will show you how to install the Jupyter notebook on your Spark cluster and use it to process Reddit comment data residing on S3. [Sign up to our mailing list](#) to be the first to get this and future tutorials.

### ***Already a data scientist or engineer?***

Join us for a two-day advanced [Apache Spark Lab](#) led by tech industry experts.

### ***Interested in transitioning to career in data engineering?***

Learn more about the [Insight Data Engineering Fellows Program](#) in New York and Silicon Valley.

Austin Ouyang

Read [more posts](#) by this author.

Share this post



Be the first to get new Data Labs tutorials.

Your email address

**SUBSCRIBE**

Insight

**DATA SCIENCE FELLOWSHIP**

**DATA ENGINEERING FELLOWSHIP**

**HEALTH DATA FELLOWSHIP**

Data Labs

**VISUALIZATION LAB**

**SPARK LAB**

**BLOG**

©2015 Insight Data Labs



[datalabs@insightdatascience.com](mailto:datalabs@insightdatascience.com)