

This is targeted documentation for getting started with Amazon Web Services. Specifically, this covers the Elastic Compute (EC2), Simple Storage Service (S3), and Elastic Map Reduce (EMR) for executing custom Hadoop programs. This is comprised of links from the AWS documentation as well as explanations & tips for a quick and successful configuration. This document also covers how to access all of these services through the AWS Command Line Interface (CLI) and the AWS SDK providing programmatic control of all AWS resources.

## Getting Started

This is the Getting Started Guide:

<http://docs.aws.amazon.com/gettingstarted/latest/awsgsg-intro/gsg-aws-intro.html>

It gives a very broad overview. The first step is to create your free account:

<http://aws.amazon.com/free/>

Make sure your info is correct as they will email you a validation link. This free account will give you free access to an EC2 micro instance (i.e., your own private machine in the cloud). More importantly, they will credit you \$100+ for use on AWS. While EC2 micro is free, storage (S3) and map-reduce (EMR) are not.

## Console

With your credentials, you should now be able to access the console:

<https://console.aws.amazon.com/console/home>

Whenever presented with a drop-down for Region of the world, pick the closest geographic area (e.g., N. Virginia).

## Security

There are two levels of security used by AWS. The first are your root access keys. Beware: "Anyone who has the [root] access key for your AWS account has unrestricted access to all the resources in your account, including billing information."

<http://docs.aws.amazon.com/general/latest/gr/managing-aws-access-keys.html>

The second level of security is for an IAM User account.

<https://console.aws.amazon.com/iam>

It is a very good policy to create an IAM Administrator account for yourself right away and use its login and key for all of your AWS web and CLI use. Full instructions for creating a new account are below. Your credentials will include an Access Key ID and a Secret Access Key. Choose Download Credentials, and store the keys in a secure location. Do not lose them.

## EC2

“Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable computing capacity—literally, servers in Amazon's data centers—that you use to build and host your software systems.”

<https://aws.amazon.com/documentation/ec2/>

This is your own virtual server. You can do anything on this virtual instance that you can on your own Linux box. Create and connect to an instance per these instructions:

[http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-launch-instance\\_linux.html](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-launch-instance_linux.html)

Make sure to pick a size within your free tier constraints. Once the machine is running, connect to it via SSH. Connection instructions are linked from the above page.

You can upgrade this machine instance with any Linux package to build your own development or testing environment. Make sure to shutdown the instance when you are done using it as there are time limits in order to stay under the free threshold.

### S3

“Amazon Simple Storage Service (Amazon S3) is storage for the Internet. You can use Amazon S3 to store and retrieve any amount of data at any time, from anywhere on the web.”

This global file storage can be used by AMR programs (e.g., Hadoop) as well as user programs executing on either your local machine or on EC2.

<https://aws.amazon.com/documentation/s3/>

For our purposes, input data will reside on S3 and EMR Hadoop programs will also use it for their logs and output. The main tasks are creating a “bucket” and a “folder” hierarchy as well as uploading and downloading files. These tasks are simple from the S3 web interface:

<https://console.aws.amazon.com/s3/>

Here is the User Guide:

<http://docs.aws.amazon.com/AmazonS3/latest/UG/Welcome.html>

Note that I am occasionally unable to delete a file or folder via the web interface. If this happens, logout and back in or preferably use the CLI (discussed below).

### EMR

“Amazon Elastic MapReduce (Amazon EMR) is a managed cluster platform that simplifies running big data frameworks, such as Apache Hadoop...”

<http://docs.aws.amazon.com/ElasticMapReduce/latest/ManagementGuide/emr-what-is-emr.html>

As with Hadoop and its related applications (e.g., Spark, Hive, Pig), this is a very large and complex service. This section will describe the steps necessary for executing a custom Hadoop task on EMR.

<https://console.aws.amazon.com/elasticmapreduce/>

1. Package your Hadoop application in a JAR and test that it works locally in a standalone configuration. It is necessary that your program take command-line parameters for the input and output paths as they will be S3 URLs on EMR.
2. Use S3 web interface to create a bucket and two subdirectories (input & log).
  - a. Upload your JAR file into the my-bucket root.
  - b. Upload your input files into the my-bucket/input folder.
3. Go to EMR Console and create a cluster.

- Name Cluster
- Set Log location to: s3://my-bucket/log
- Select "Launch mode" of "Step execution"
- Select "Step type" to "Custom JAR"
- Click configure button. Select the JAR. Assuming your main program accepts input and output folders as parameters, set the Hadoop program command line arguments to include your:
  - job class
  - s3://my-bucket/input
  - s3://my-bucket/output
- Choose the latest release of emr stack (different than graphic below).
- The remaining defaults are fine for now. Click Create Cluster.

## Create Cluster - Quick Options [Go to advanced options](#)

### General Configuration

Cluster name

☒ Logging ⓘ

S3 folder

Launch mode ☐ Cluster ⓘ ☒ Step execution ⓘ

### Add steps

A step is a unit of work submitted to an application running on your EMR cluster. EMR programmatically installs the applications needed to execute the added steps. [Learn more](#)

Name	Action on failure	JAR location	Arguments	
Custom JAR	Terminate cluster	s3://my-bucket/job.jar	package.MyJob s3://my-bucket/input s3://my-bucket/output	

Step type

### Software configuration

Vendor ☒ Amazon ☐ MapR

Release  ⓘ

Applications Hadoop 2.7.2 ⓘ

### Hardware configuration

Instance type  ⓘ The selected instance type adds a default 32 GIB GP2 EBS volume per instance. [Learn more](#)

Number of instances  (1 master and 2 core nodes)

### Security and access

Permissions ☒ Default ☐ Custom

Use default IAM roles. If roles are not present, they will be automatically created for you with managed policies for automatic policy updates.

EMR role [EMR\\_DefaultRole](#) ⓘ

EC2 instance profile [EMR\\_EC2\\_DefaultRole](#) ⓘ

[Cancel](#)

The new cluster is visible in the Cluster List. It will take a few minutes to be provisioned but execute quickly once it spins up. The program output will be in its directory but likely separated into multiple files due to the reducers executing on different machines in the cluster.

### AWS Command Line Interface (CLI)

“The AWS Command Line Interface is a unified tool to manage your AWS services. With just one tool to download and configure, you can control multiple AWS services from the command line and automate them through scripts.”

<http://docs.aws.amazon.com/cli/latest/userguide/cli-chap-welcome.html>

Install:

<http://docs.aws.amazon.com/cli/latest/userguide/installing.html#install-bundle-other-os>

Configure:

<http://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting-started.html>

The rest of this section will detail the steps to:

- Create IAM account
- Assign Administrator privileges to new account
- Download keys for this account
- Install CLI
- Configure CLI
- Execute CLI commands

### Create IAM Account

For security purposes, it is important to create an IAM account for working with AWS services rather than using your “root” account created at sign-up.


1. IAM service -> select Users from left menu.
2. Click Add User button at top.
3. Enter your chosen User Name in the top field.
4. Make sure to select Programmatic access and AWS Management Console access checkboxes in the Access Type section.
  - a. Choose a Custom Password
  - b. Deselect Require Change Password so you can keep your password.
5. Click button Next: Permissions

### Assign Administrator privileges to new account


Administrator privileges will allow access and use of the AWS services but prevent access to billing and general account maintenance. This is desirable and you should use your original root email address and password for those purposes.

6. Although you may create an Administrator group and assign the new account to it, it is simplest to directly give the new account an Administrator policy. Choose:  
Attach existing policies directly
7. Type Admin partial string to filter the policies in the list below.
8. Select Administrator Access, as shown in the figure below:


Set permissions for Rob



Add user to group




Copy permissions from existing user







Attach existing policies directly

Attach one or more existing policies directly to the user or create a new policy. [Learn more](#)

Create policy  Refresh

Filter: Policy type

	Policy name	Type	Attachments	Description
<input type="checkbox"/>	 DatabaseAdministrator	Job function	0	Grants full access permissions to AWS services and a
<input type="checkbox"/>	 ServiceCatalogAdminFullAccess	AWS managed	0	Provides full access to the service catalog admin cons
<input type="checkbox"/>	 SystemAdministrator	Job function	0	Grants full access permissions necessary for resource
<input checked="" type="checkbox"/>	 AdministratorAccess	Job function	1	Provides full access to AWS services and resources.

9. Click button Next: Review
10. Your review screen should look like figure below, except for User name (of course):

Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

User details


User name	Rob
AWS access type	Programmatic access and AWS Management Console access
Console password type	Custom
Require password reset	No

Permissions summary

The following policies will be attached to the user shown above.


Type	Name
Managed policy	<a href="#">AdministratorAccess</a>


11. Click button Create user
12. You will see completion screen, as shown in figure below:




 **Success**

You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at <https://your-account-id.signin.aws.amazon.com/console>

 Download .csv



	User	Access key ID	Secret access key	Email login instructions
	 Rob	AKIAIY6ARWS4FWG26RKA	***** <a href="#">Show</a>	<a href="#">Send email</a> 

Close

Download keys for this account

13. Make sure to immediately download .csv with your new user account's access keys by clicking the Download .csv button.
14. Bookmark the URL highlighted above. This should become your login for AWS console going forward. Use your new User Name and Password you just created as credentials.

Install CLI

15. From a terminal shell (\$ prompt):

- a. `$ cd ~/Downloads`
- b. `$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-bundle.zip"`
- c. `$ unzip awscli-bundle.zip`
- d. `$ sudo ./awscli-bundle/install -i /usr/local/aws -b /usr/local/bin/aws`
- e. Answer affirmatively to the install program queries.

Configure CLI

16. From a terminal shell (\$ prompt):

- a. `$ cd`
- b. Assuming your credentials.csv file downloaded in step 13 is in ~/Downloads:  
`$ cat ~/Downloads/credentials.csv`
- c. Substituting the keys in your credentials.csv file:  
`$ aws configure`
  - i. AWS Access Key ID [None]: <your-access-key-id>
  - ii. AWS Secret Access Key [None]: <your-secret-access-key>
  - iii. Default region name [None]: us-east-1
  - iv. Default output format [None]: text
- d. The above step should create two secure files in your ~/.aws directory.  
`$ ls -al ~/.aws`

```
drwx----- 2 joe joe 4096 Nov  1 21:49 .
drwxr-xr-x 34 joe joe 4096 Jan 20 19:59 ..
-rw----- 1 joe joe  43 Nov  1 21:48 config
-rw----- 1 joe joe 116 Nov  1 21:49 credentials
```
- e. Check that they contain the correct information from configure step above:  
`$ cat ~/.aws/credentials`  
`$ cat ~/.aws/config`

Execute CLI Commands:

17. CLI S3 command to check that configuration works:

`$ aws s3 ls s3://<your-bucket-name>`

Note that the bucket must have been previously created in the same region that you listed in the configuration step above (e.g., us-east-1). You may check by logging into the console, selecting the US East (N. Virginia) region and choosing S3 service. All of your buckets are listed there.

18. The web services each provide extensive CLI commands. These are a few useful ones:

- a. Make bucket:

```
$ aws s3 mb s3://my-bucket
```

- b. Upload file:

```
$ aws s3 cp my-jar.jar s3://my-bucket/
```

- c. Delete folder:

```
$ aws s3 rm s3://my-bucket/output/ --recursive
```

- d. Create cluster.

Note that you should create a cluster to execute your MapReduce program via the AWS EMR web interface first. This will give you experience with this UI and allow you to easily gather configuration information needed for the CLI example. Make sure to customize the red configuration parameters before executing:

```
$ aws emr create-cluster \
    --name "WordCount Cluster" \
    --release-label emr-5.2.1 \
    --instance-groups \
' [{"InstanceCount":2,"InstanceGroupType":"CORE","InstanceType":"m4.large"}, {"InstanceCount":1,"InstanceGroupType":"MASTER","InstanceType":"m4.large"} ]' \
    --applications Name=Hadoop \
    --steps \
' [{"Args":["your.package.MainClass","s3://my-bucket/input","s3://my-bucket/output"],"Type":"CUSTOM_JAR","Jar":"s3://my-bucket/my-jar.jar","ActionOnFailure":"TERMINATE_CLUSTER","Name":"Custom JAR"} ]' \
    --log-uri s3://my-bucket/log \
    --service-role EMR_DefaultRole \
    --ec2-attributes InstanceProfile=EMR_EC2_DefaultRole,SubnetId=your-subnet-id \
    --region us-east-1 \
    --enable-debugging \
    --auto-terminate
```

- e. Download output, following cluster termination:

```
$ mkdir output
```

```
$ aws s3 sync s3://my-bucket/output output
```

## AWS Java SDK

“The AWS SDK for Java provides a Java API for AWS infrastructure services. Using the SDK, you can build applications on top of Amazon S3, Amazon EC2, Amazon DynamoDB, and more.”

<https://aws.amazon.com/documentation/sdk-for-java/>

There are SDKs for many other languages and even different platforms (e.g., Android). AWS access greatly extends the capabilities of applications. This section will use the SDK to access EC2, S3 and EMR.

Download the Java AWS SDK:

<http://sdk-for-java.amazonwebservices.com/latest/aws-java-sdk.zip>

or, preferably, use Maven to manage this and all of your dependencies:

```
<dependency>
```

```
  <groupId>com.amazonaws</groupId>
```

```
  <artifactId>aws-java-sdk</artifactId>
```

```
  <version>1.11.31</version>
```

</dependency>

Note: Check Maven and use latest version.

## Security

As with CLI, it is very important to get your credentials properly configured. There are many options for this but, if you configured your Access Key ID and Secret Access Key as described in the security section above, the SDK should seamlessly pick it up your credentials in your .aws directory. If not, check that you created an IAM role and downloaded id/key:

<http://docs.aws.amazon.com/general/latest/gr/managing-aws-access-keys.html>

## Example - Create EC2 Instance

This will create a new EC2 instance based on EC2 public image ami-60b6c60a and micro instance within the free tier. If you want a larger (priced) mage, check here:

<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/finding-an-ami.html>

With security properly configured, the default credential search should work. Note that you must supply your key and a configured security group. A security group limits the IP address that can connect to the EC2 instance.

```
-----
AWSCredentials credentials = new ProfileCredentialsProvider().getCredentials();
AmazonEC2Client amazonEC2Client = new AmazonEC2Client(credentials);
amazonEC2Client.setEndpoint("ec2.us-east-1.amazonaws.com");
RunInstancesRequest runInstancesRequest = new RunInstancesRequest();
runInstancesRequest.withImageId("ami-60b6c60a")
    .withInstanceType("t2.micro")
    .withMinCount(1)
    .withMaxCount(1)
    .withKeyName("YourKey")
    .withSecurityGroups("YourSG");
RunInstancesResult runInstancesResult = amazonEC2Client.runInstances(runInstancesRequest);
System.out.println("Id: " + runInstancesResult.getReservation().getInstances().get(0).getInstanceId());
-----
```

## Example - Stop EC2 Instance

```
-----
// Same credentials & client as above.
StopInstancesRequest stopInstancesRequest = new StopInstancesRequest(Arrays.asList(new String[]{instanceId}));
StopInstancesResult stopInstancesResult = amazonEC2Client.stopInstances(stopInstancesRequest);
-----
```

## Example - Start Existing EC2 Instance

```
-----
StartInstancesRequest startInstancesRequest = new StartInstancesRequest()
    .withInstanceIds(instanceId);
StartInstancesResult startInstancesResult = amazonEC2Client.startInstances(startInstancesRequest);
-----
```



## Example - S3 Tasks

```
-----
// Same credentials as above.
AmazonS3Client s3 = new AmazonS3Client(credentials);
Region usEast2 = Region.getRegion(Regions.US_EAST_1);
s3.setRegion(usEast2);
// Create bucket.
s3.createBucket(bucketName);
// List buckets.
for (Bucket bucket : s3.listBuckets()) {
    System.out.println(bucket.getName());
}
// Upload file to bucket.
s3.putObject(new PutObjectRequest(bucketName, key, aFile));
// List object in bucket.
ObjectListing objectListing = s3.listObjects(new ListObjectsRequest()
    .withBucketName(bucketName)
    .withPrefix("My"));
for (S3ObjectSummary objectSummary : objectListing.getObjectSummaries()) {
    System.out.println(objectSummary.getKey() + " " + "(size = " + objectSummary.getSize() + ")");
}
// Download and read an object (text file).
S3Object object = s3.getObject(new GetObjectRequest(bucketName, key));
BufferedReader reader = new BufferedReader(new InputStreamReader(object.getObjectContent()));
String line;
while ((line = reader.readLine()) != null) {
    System.out.println(line);
}
// Delete object.
s3.deleteObject(bucketName, key);
// Delete bucket.
s3.deleteBucket(bucketName);
-----
```

## AWS SDK Managing EMR Cluster and Steps

This API allows access to all of the EMR functionality. This is good documentation:

<http://docs.aws.amazon.com/ElasticMapReduce/latest/ManagementGuide/emr-common-programming-concepts.html>

## Example - EMR Create Cluster

```
-----
AWSCredentials credentials = new ProfileCredentialsProvider().getCredentials();
AmazonElasticMapReduce client = new AmazonElasticMapReduceClient(credentials);
HadoopJarStepConfig hadoopConfig = new HadoopJarStepConfig()
    .withJar("s3://bucket/YourJar.jar")
    .withMainClass("YourMainClass")
```

```

        .withArgs("s3://bucket/input", "s3://bucket/output");
StepConfig customStep = new StepConfig("Step Name", hadoopConfig);
RunJobFlowRequest request = new RunJobFlowRequest()
    .withName("Cluster Name")
    .withReleaseLabel("emr-4.3.0")
    .withSteps(customStep, customStep2)
    .withLogUri("s3://bucket/logs")
    .withServiceRole("EMR_DefaultRole")
    .withJobFlowRole("EMR_EC2_DefaultRole")
    .withInstances(new JobFlowInstancesConfig().withKeepJobFlowAliveWhenNoSteps(true)
        .withEc2KeyName("YourKey")
        .withInstanceCount(3)
        .withKeepJobFlowAliveWhenNoSteps(true)
        .withMasterInstanceType("m3.xlarge")
        .withSlaveInstanceType("m3.xlarge"));
RunJobFlowResult result = client.runJobFlow(request);
System.out.println("jobFlowId: " + result.getJobFlowId());
// Add step to running cluster.
HadoopJarStepConfig hadoopConfig2 = new HadoopJarStepConfig()
    .withJar("s3://bucket/YourJar.jar")
    .withMainClass("YourMainClass")
    .withArgs("s3://bucket/input", "s3://bucket/output");
StepConfig customStep2 = new StepConfig("Step Name", hadoopConfig2);
AddJobFlowStepsRequest request2 = new AddJobFlowStepsRequest()
    .withJobFlowId(jobFlowId)
    .withSteps(customStep3);
AddJobFlowStepsResult result2 = client.addJobFlowSteps(request);

```