Spark Lightning-fast cluster computing

Spark doesn't need an introduction, but just in case, it extends Hadoop's distributed / parallel computing paradigm by using both live memory in addition to disk storage. This makes for a much faster tool. As of version 1.4, **SparkR** is included in Apache's Spark build. We'll be using version 1.5 in this walk-through. Here we'll create our **AWS EC2** instance.

- AWS Instance
- VPC
- EC2
- Connecting to EC2

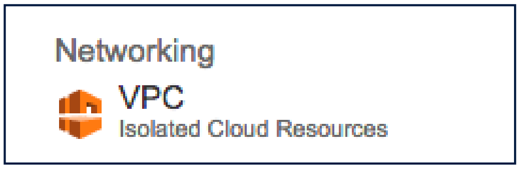
Cluster Launcher

AWS Instance

In order to approach this from the same vantage point, we'll use a small EC2 instance to launch our Spark clusters. You will need an amazon **AWS** account and the ability to **Secure Shell Tunneling (SSH)** into AWS (more on this later). Keep in mind that AWS instances cost money and the more clusters you need, the more money Amazon will charge. So, always remember to "Stop" or "Terminate" you instances when not needed! First, sign into the AWS Console:

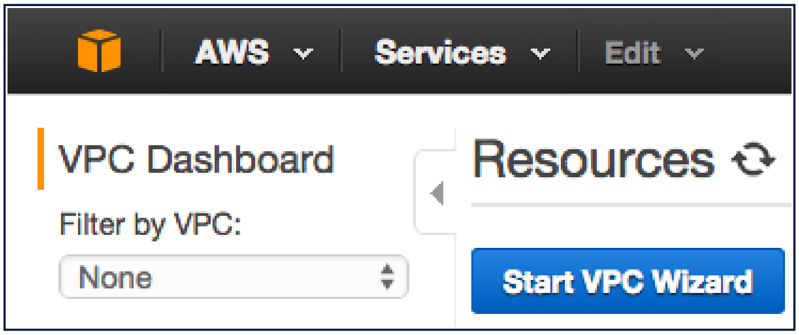
VPC

Under header Networking, select VPC:

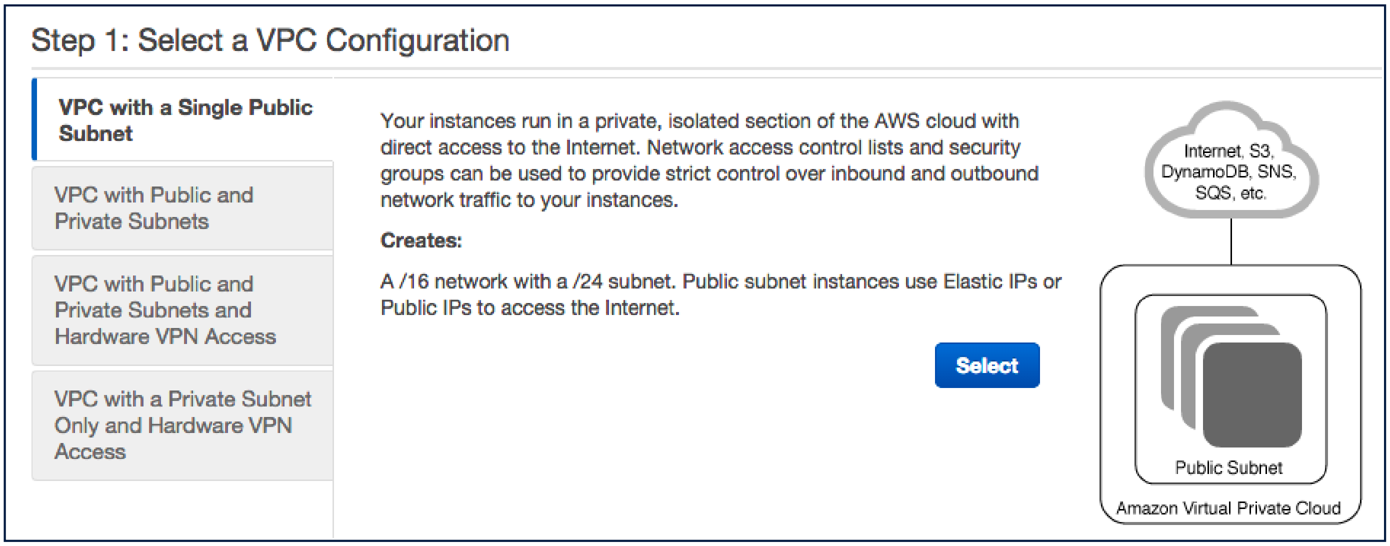


A **virtual private cloud (VPC)** will determine who and what gets to access the site. We will use the wizard and content ourselves with only on VPC. In an enterprise-level application, you will want at least 4, 2 to be private and run your database, and two to be public and hold your web-serving application. By duplicating the private and public VPCs you can benefit from fail-over and load balancing tools. For our purposes, we'll just go with one public VPC:

Start the "wizard":



Start the wizard and select "VPC with a Single Public Subnet":



Step 1: Select a VPC Configuration

VPC with a Single Public Subnet

VPC with Public and Private Subnets

VPC with Public and Private Subnets and Hardware VPN Access

VPC with a Private Subnet Only and Hardware VPN Access

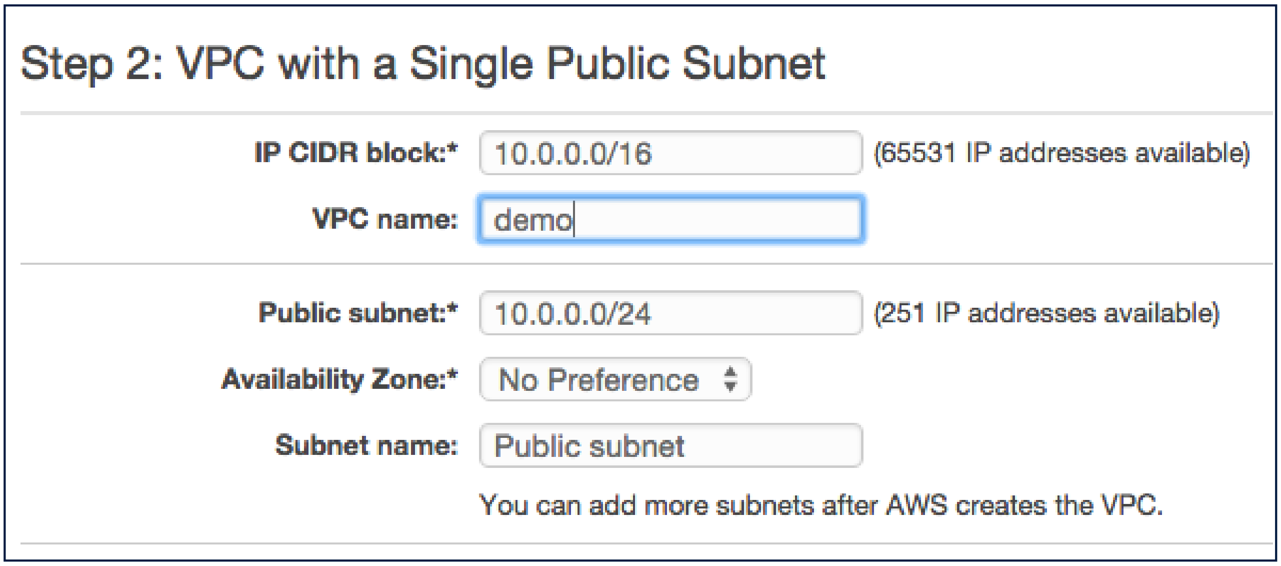
Your instances run in a private, isolated section of the AWS cloud with direct access to the Internet. Network access control lists and security groups can be used to provide strict control over inbound and outbound network traffic to your instances.

Creates:

A /16 network with a /24 subnet. Public subnet instances use Elastic IPs or Public IPs to access the Internet.

Select
Internet, S3, DynamoDB, SNS, SQS, etc.
Public Subnet
Amazon Virtual Private Cloud

Most of the defaults are fine except you should add a name under “VPC name”:



Step 2: VPC with a Single Public Subnet

IP CIDR block:* 10.0.0.0/16 (65531 IP addresses available)

VPC name: demo

Public subnet:* 10.0.0.0/24 (251 IP addresses available)

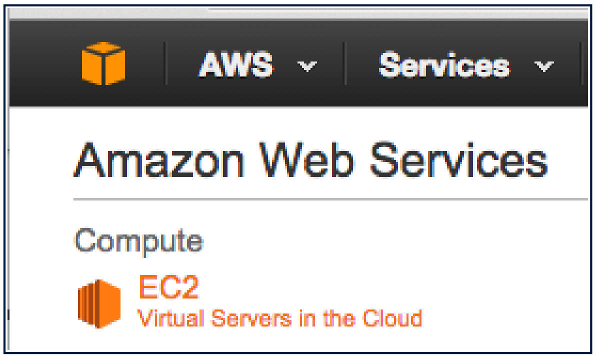
Availability Zone:* No Preference

Subnet name: Public subnet

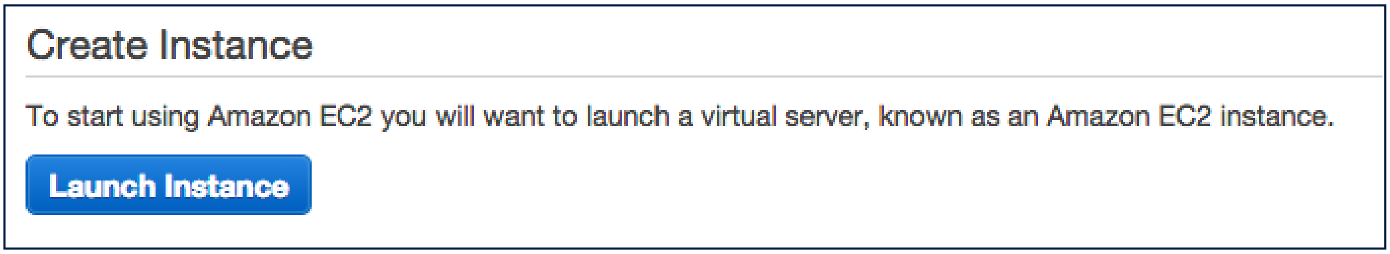
You can add more subnets after AWS creates the VPC.

EC2

VPC is done, let’s now create our EC2 instance - this is going to be our cluster-launching machine. Click on the orange cube in the upper left corner of the page. From the ensuing menu, choose the first option, “EC2”:



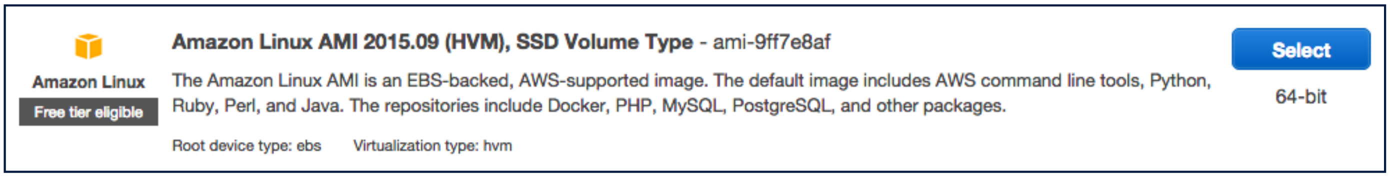
In `Create Instance`, select `Launch Instance`:



Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Select the first option `Amazon Linux AMI`:



Amazon Linux

Free tier eligible

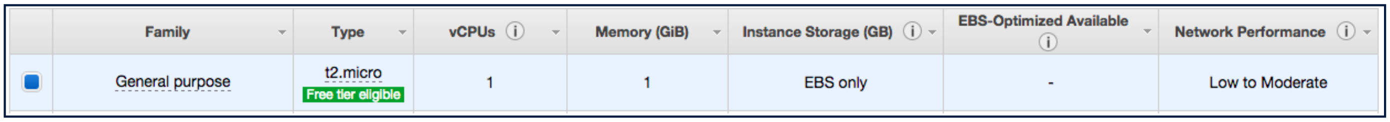
Amazon Linux AMI 2015.09 (HVM), SSD Volume Type - ami-9ff7e8af

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

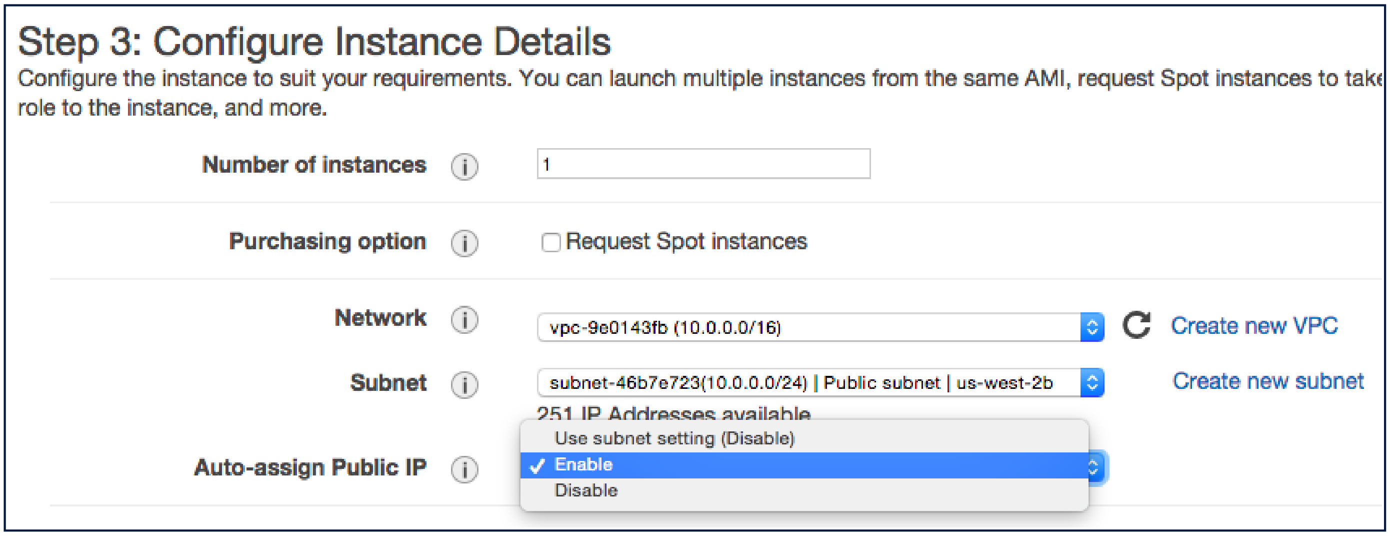
64-bit

In **Step 2**, continue with the preselected machine and click `Next: Configure Instance Details`:



	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
■	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate

In **Step 3**, keep all defaults but change the `Auto-assign Public IP` to `Enable`:

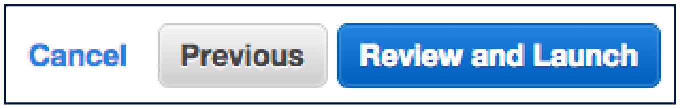


Step 3: Configure Instance Details

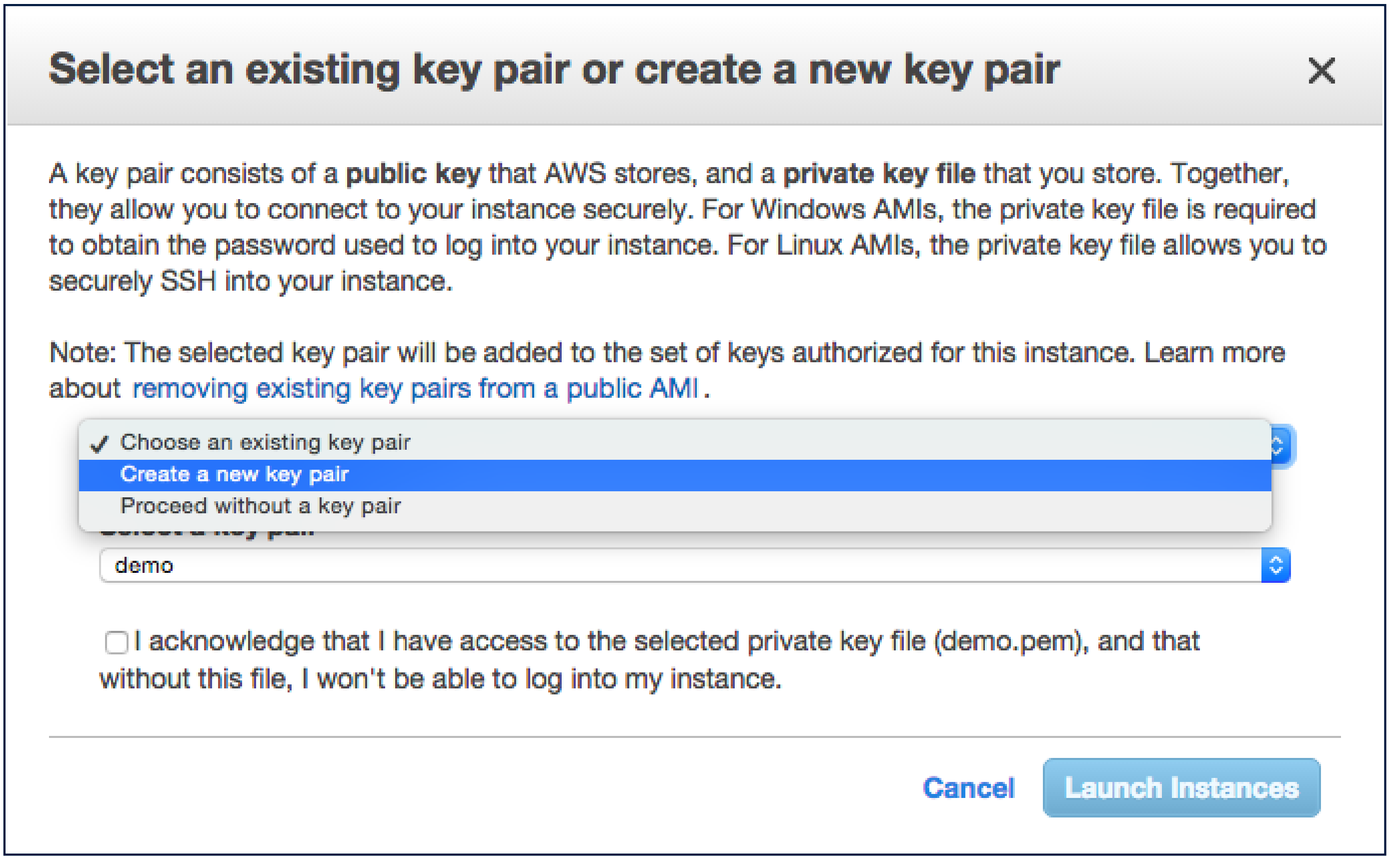
Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take role to the instance, and more.

Number of instances	1
Purchasing option	□ Request Spot instances
Network	vpc-9e0143fb (10.0.0.0/16)
Subnet	subnet-46b7e723(10.0.0.0/24) | Public subnet | us-west-2b 251 IP Addresses available
Auto-assign Public IP	Use subnet setting (Disable) ✓ Enable Disable

Select `Review and Launch`:



In **Step 7**, `SSH port 22` is opened by default so there is nothing for us to do but to click `Launch`. It will open a pop-up box where you will need to create a new key pair:



Select an existing key pair or create a new key pair

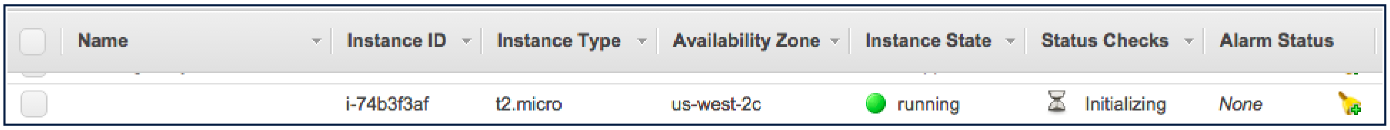
A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI .

□ I acknowledge that I have access to the selected private key file (demo.pem), and that without this file, I won't be able to log into my instance.

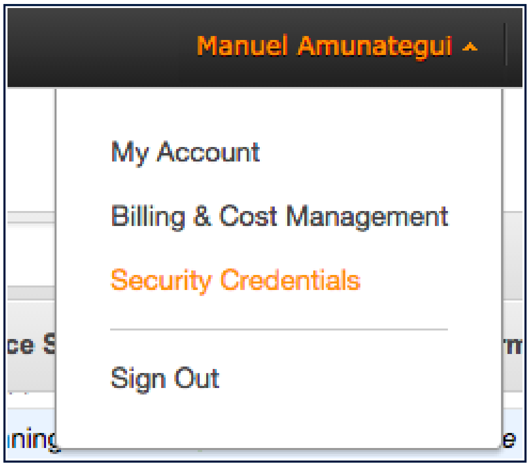
Key-pair is a security file that will live on your machine and is required to SSH into the instance. I tend to create them and leave them in my downloads. What ever you decided to do, make sure you know where it is as you'll need to pass a path to it every time you want to connect to it. Create a new one, check the acknowledgment check-box.

Finally, click Launch Instances. That's it for our instance. Hit the View Instances on the next page as the machine is being initialized:

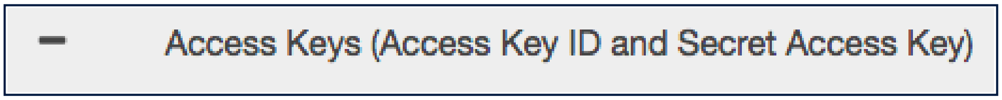


□	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status
□		i-74b3f3af	t2.micro	us-west-2c	● running	⌛ Initializing	*None*

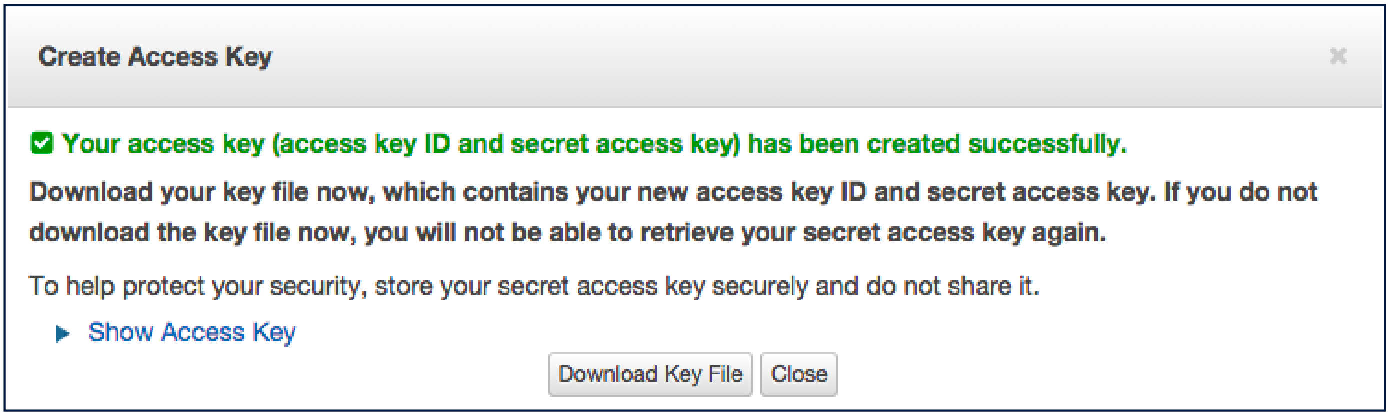
We need to wait till the Initializing is done. Meanwhile we can get the security credentials that will be needed to launch the clusters. Go to your account name drop down in the top right corner and click `Security Credentials`:



Choose Access Keys



And Create Access Key



Create Access Key

Your access key (access key ID and secret access key) has been created successfully.

Download your key file now, which contains your new access key ID and secret access key. If you do not download the key file now, you will not be able to retrieve your secret access key again.

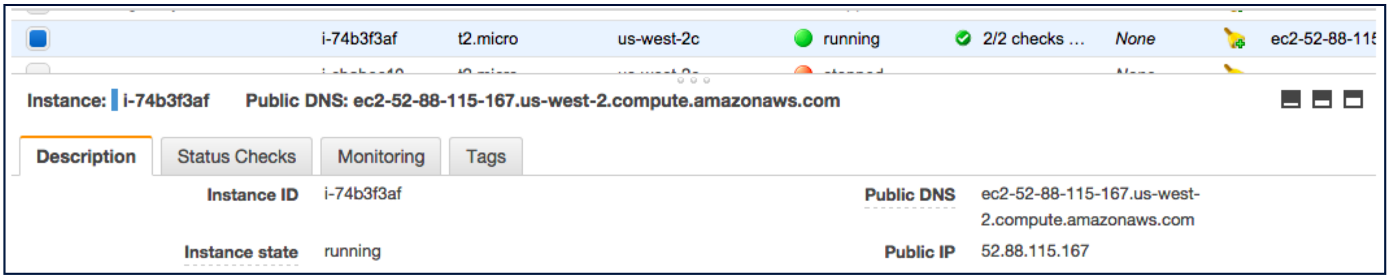
To help protect your security, store your secret access key securely and do not share it.

This will download two strings:

```
AWS_ACCESS_KEY_ID=...
AWS_SECRET_ACCESS_KEY= ...
```

Keep that download handy as we'll need the values shortly. If you need more information on security keys check out this Amazon blog post

Back at the instance window, check that our new instance is running, you can click on the check box to access the assigned dynamic IP address:



■	i-74b3f3af	t2.micro	us-west-2c	● running	✓ 2/2 checks ...	*None*	ec2-52-88-115

Instance: i-74b3f3af Public DNS: ec2-52-88-115-167.us-west-2.compute.amazonaws.com

Instance ID: i-74b3f3af

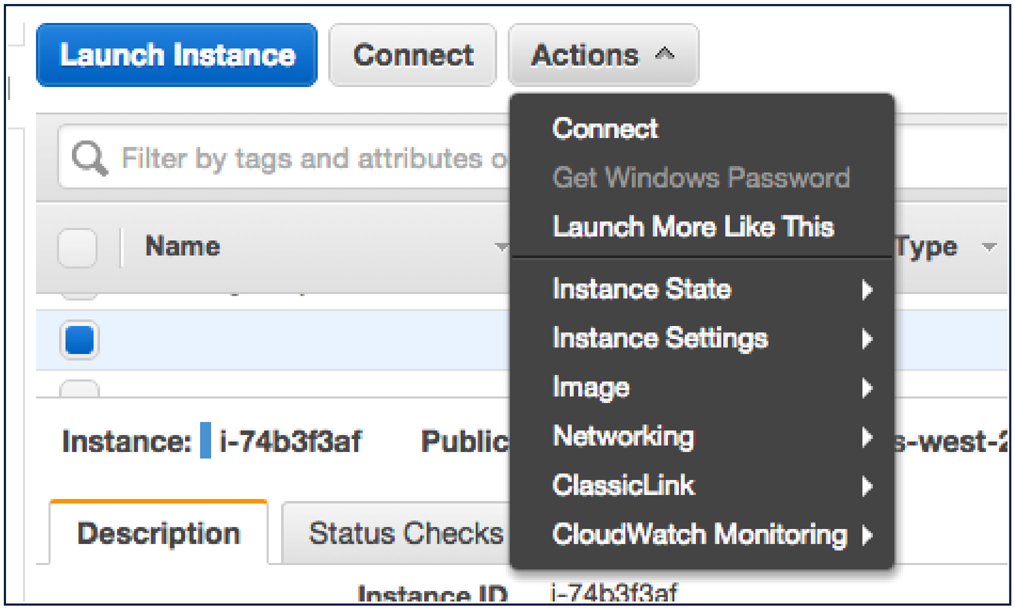
Instance state: running

Public DNS: ec2-52-88-115-167.us-west-2.compute.amazonaws.com

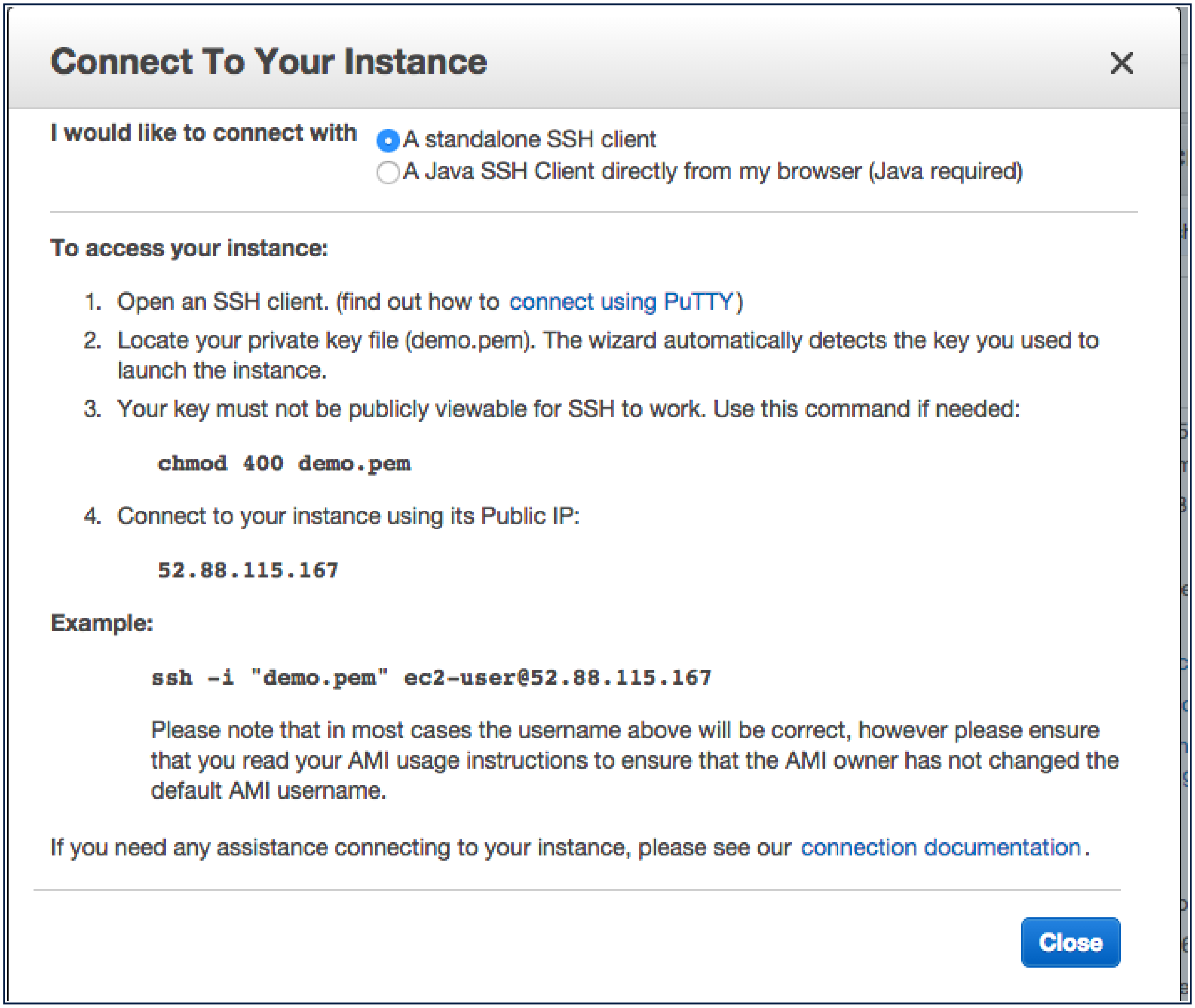
Public IP: 52.88.115.167

Connecting to EC2

Use the `Actions` button to get the exact SSH string to reach the instance:



And select `Connect`:



Connect To Your Instance

I would like to connect with ● A standalone SSH client
○ A Java SSH Client directly from my browser (Java required)

To access your instance:

1. Open an SSH client. (find out how to connect using PuTTY)
2. Locate your private key file (demo.pem). The wizard automatically detects the key you used to launch the instance.
3. Your key must not be publicly viewable for SSH to work. Use this command if needed:

```
chmod 400 demo.pem
```

4. Connect to your instance using its Public IP:

```
52.88.115.167
```

Example:

```
ssh -i "demo.pem" ec2-user@52.88.115.167
```

Please note that in most cases the username above will be correct, however please ensure that you read your AMI usage instructions to ensure that the AMI owner has not changed the default AMI username.

If you need any assistance connecting to your instance, please see our connection documentation.

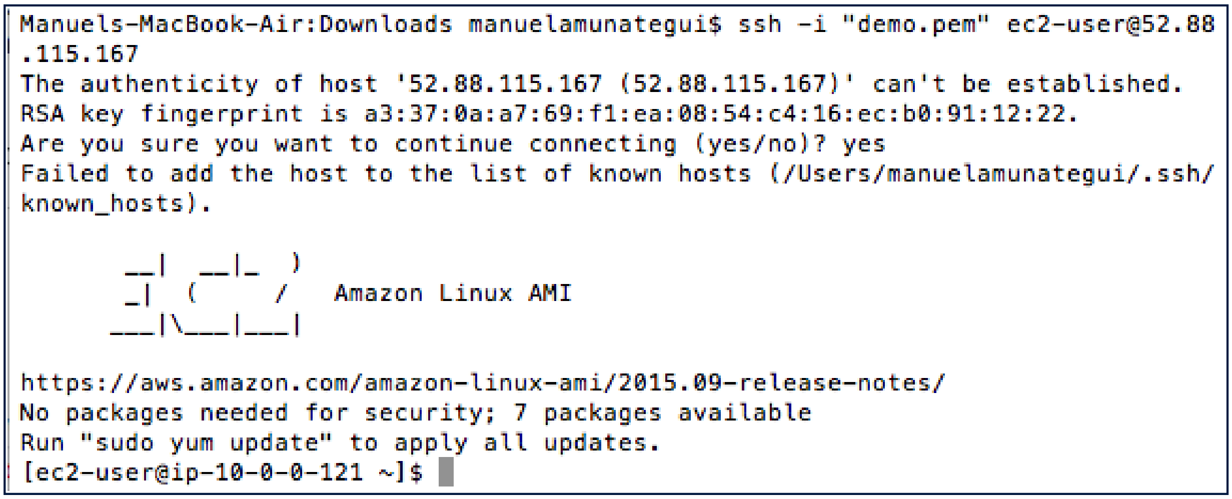
Read the instruction on the pop-up box. The last line states your connection string: `ssh -i "udemy.pem" ec2-user@52.88.115.167`. To use it on the Mac, open your terminal and navigate to your Downloads folder (or wherever you moved your pem key-pair file). As per instructions, apply `chmod 400 udemy.pem` and copy the example string.

SSH

Paste it into the terminal and follow the instructions to log into your EC2 instance:

```
ssh -i "udemy.pem" ec2-user@52.88.115.167
```

Your terminal should confirm the EC2:



```
Manuels-MacBook-Air:Downloads manuelamunategui$ ssh -i "demo.pem" ec2-user@52.88
.115.167
The authenticity of host '52.88.115.167 (52.88.115.167)' can't be established.
RSA key fingerprint is a3:37:0a:a7:69:f1:ea:08:54:c4:16:ec:b0:91:12:22.
Are you sure you want to continue connecting (yes/no)? yes
Failed to add the host to the list of known hosts (/Users/manuelamunategui/.ssh/
known_hosts).

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2015.09-release-notes/
No packages needed for security; 7 packages available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-10-0-0-121 ~]$ 
```

Next we will get **Spark clusters** up and running.