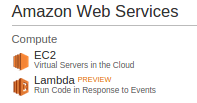
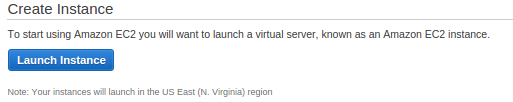
**[Docker install on Amazon Linux AMI](http://www.bogotobogo.com/DevOps/Docker/Docker_Install_On_Amazon_Linux_AMI.php)**

**Install on Amazon Linux AMI**

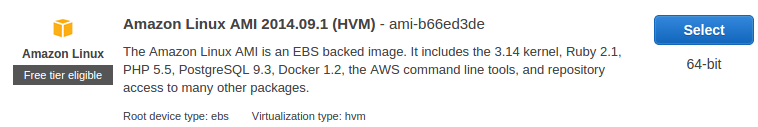
1. We need an [AWS account (http://aws.amazon.com/)](http://aws.amazon.com/).
2. Choose EC2 from [Amazon Web Services](https://console.aws.amazon.com/console/home?region=us-east-1) Console.



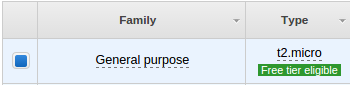
1. From [EC2 Dashboard](https://console.aws.amazon.com/ec2/v2/home?region=us-east-1), clock on Launch Instance:



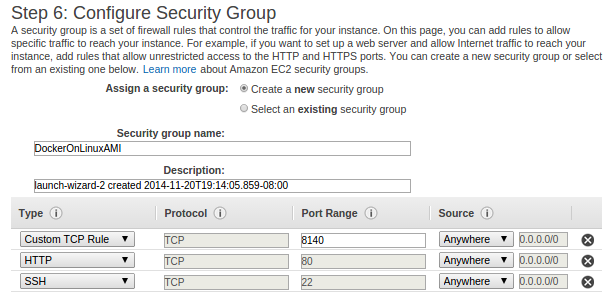
1. On the [Choose an Amazon Machine Image (AMI)](https://console.aws.amazon.com/ec2/v2/home?#LaunchInstanceWizard:) menu on the AWS Console. Click the Select button for a 64Bit (Amazon Linux AMI)

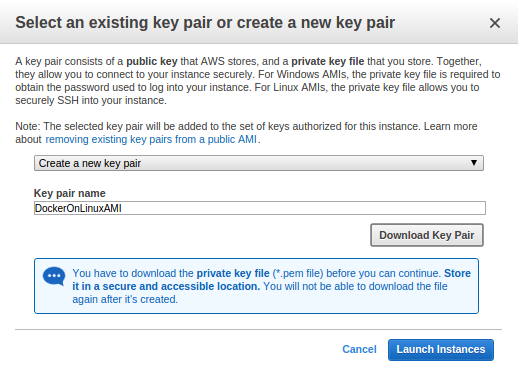


1. For testing we can use the default (possibly free) t2.micro instance (more info on [pricing](http://aws.amazon.com/ec2/pricing/)).



1. We may need to set up our Security Group to allow **SSH**. By default all incoming ports to our new instance will be blocked by the AWS Security Group, so we might just get timeouts when we try to connect.



1. Creating a new key pair:
2. After a few more standard choices where defaults are probably ok, our AWS Ubuntu instance with Docker should be running!

Running_Linux_AMI_Docker.png

**ssh to EC2**

Amazon Linux does not allow remote root SSH by default. Also, password authentication is disabled to prevent brute-force password attacks. To enable SSH logins to an Amazon Linux instance, we must provide our key pair (DockerOnLinuxAMI.pem) to the instance at launch. We must also set the security group (DockerOnLinuxAMI) used to launch our instance to allow SSH access. By default, the only account that can log in remotely using SSH is **ec2-user**.

**We need to make the key file read-only:**

k@laptop:~$ chmod 400 /home/k/Downloads/DockerOnEC2.pem

Then, ssh to Linux AMI account with **ec2-user** as our user name:

k@laptop:~$ ssh -i DockerOnLinuxAMI.pem ec2-user@ec2-54-174-11-166.compute-1.amazonaws.com

\_\_| \_\_|\_ )

\_| ( / Amazon Linux AMI

\_\_\_|\\_\_\_|\_\_\_|

https://aws.amazon.com/amazon-linux-ami/2014.09-release-notes/

10 package(s) needed for security, out of 24 available

Run "sudo yum update" to apply all updates.

[ec2-user@ip-172-31-53-242 ~]$

Let's digress a bit here. The ssh command is too long, and we do not want to type the long command (ssh -i <path to private key> ec2-user@<public IP address>) every time we do ssh. So, let's try make it short. We want to create a new file **~/.ssh/config**:

Host ami

Hostname ec2-54-174-11-166.compute-1.amazonaws.com

User ec2-user

IdentityFile ~/.ssh/DockerOnLinuxAMI.pem

To make that work, we need to move **DockerOnLinuxAMI.pem** file to **~/.ssh**. First, we should logout from EC2:

[ec2-user@ip-172-31-53-242 ~]$ exit

Thenn, on our local machine, we move the key file:

k@laptop:~$ mv ~/Downloads/DockerOnLinuxAMI.pem ~/.ssh/

Then, try to ssh into Linux AMI EC2 again:

k@laptop:~$ ssh ami

Last login: Fri Nov 21 04:02:58 2014 from 108-239-135-40.lightspeed.frokca.sbcglobal.net

\_\_| \_\_|\_ )

\_| ( / Amazon Linux AMI

\_\_\_|\\_\_\_|\_\_\_|

https://aws.amazon.com/amazon-linux-ami/2014.09-release-notes/

10 package(s) needed for security, out of 24 available

Run "sudo yum update" to apply all updates.

[ec2-user@ip-172-31-53-242 ~]$

Now we can simply type in **ssh ami** to login to AMI EC2 instance!

Let's update to make sure our source list is up-to-date:

[ec2-user@ip-172-31-53-242 ~]$ sudo yum update

**Docker install on Linux AMI**

[ec2-user@ip-172-31-53-242 ~]$ sudo yum install -y docker

Loaded plugins: priorities, update-motd, upgrade-helper

Resolving Dependencies

--> Running transaction check

---> Package docker.x86\_64 0:1.3.1-1.0.amzn1 will be installed

--> Processing Dependency: libcgroup for package: docker-1.3.1-1.0.amzn1.x86\_64

--> Running transaction check

---> Package libcgroup.x86\_64 0:0.40.rc1-5.11.amzn1 will be installed

--> Finished Dependency Resolution

Dependencies Resolved

===============================================================================================

Package Arch Version Repository Size

===============================================================================================

Installing:

docker x86\_64 1.3.1-1.0.amzn1 amzn-updates 5.1 M

Installing for dependencies:

libcgroup x86\_64 0.40.rc1-5.11.amzn1 amzn-main 146 k

Transaction Summary

===============================================================================================

Install 1 Package (+1 Dependent package)

Total download size: 5.2 M

Installed size: 18 M

Downloading packages:

(1/2): docker-1.3.1-1.0.amzn1.x86\_64.rpm | 5.1 MB 00:00

(2/2): libcgroup-0.40.rc1-5.11.amzn1.x86\_64.rpm | 146 kB 00:00

-----------------------------------------------------------------------------------------------

Total 40 MB/s | 5.2 MB 00:00:00

Running transaction check

Running transaction test

Transaction test succeeded

Running transaction

Installing : libcgroup-0.40.rc1-5.11.amzn1.x86\_64 1/2

Installing : docker-1.3.1-1.0.amzn1.x86\_64 2/2

Verifying : libcgroup-0.40.rc1-5.11.amzn1.x86\_64 1/2

Verifying : docker-1.3.1-1.0.amzn1.x86\_64 2/2

Installed:

docker.x86\_64 0:1.3.1-1.0.amzn1

Dependency Installed:

libcgroup.x86\_64 0:0.40.rc1-5.11.amzn1

Complete!

To check the version though we can see from the output above:

[ec2-user@ip-172-31-53-242 ~]$ docker -v

Docker version 1.3.1, build c78088f/1.3.1

**Running Docker**

We can run Docker with service docker start command:

[ec2-user@ip-172-31-53-242 ~]$ sudo service docker start

Starting cgconfig service: [ OK ]

Starting docker: [ OK ]

[ec2-user@ip-172-31-53-242 ~]$ ps aux|grep docker

root 6744 0.1 0.8 283520 8880 pts/0 Sl 05:44 0:00 /usr/bin/docker -d

ec2-user 6909 0.0 0.0 110284 836 pts/0 S+ 05:45 0:00 grep docker

[ec2-user@ip-172-31-53-242 ~]$ sudo service docker status

docker (pid 6744) is running...