CFN Cluster

Tuesday, February 9, 2016 4:44 PM

INTRO

This pdf is hosted at http://djargon.azurewebsites.net/pdf/Doc02_AzureCloudFormationNetworkClusterBasics.pdf.

It is a (circa 2016) step-by-step with screencaps for bringing up an AWS cluster for parallel computing. This procedural uses the 'Cloud Formation Network' or CFN technology available from AWS via GitHub. It is a 'commonly used template' and I imagine that (because there are a number of steps) the experience will be improved over time.

IT USES

AWS, primarily the console, and Linux.

YOU WILL NEED

An AWS account.

QUALIFIERS

Account information has been redacted.

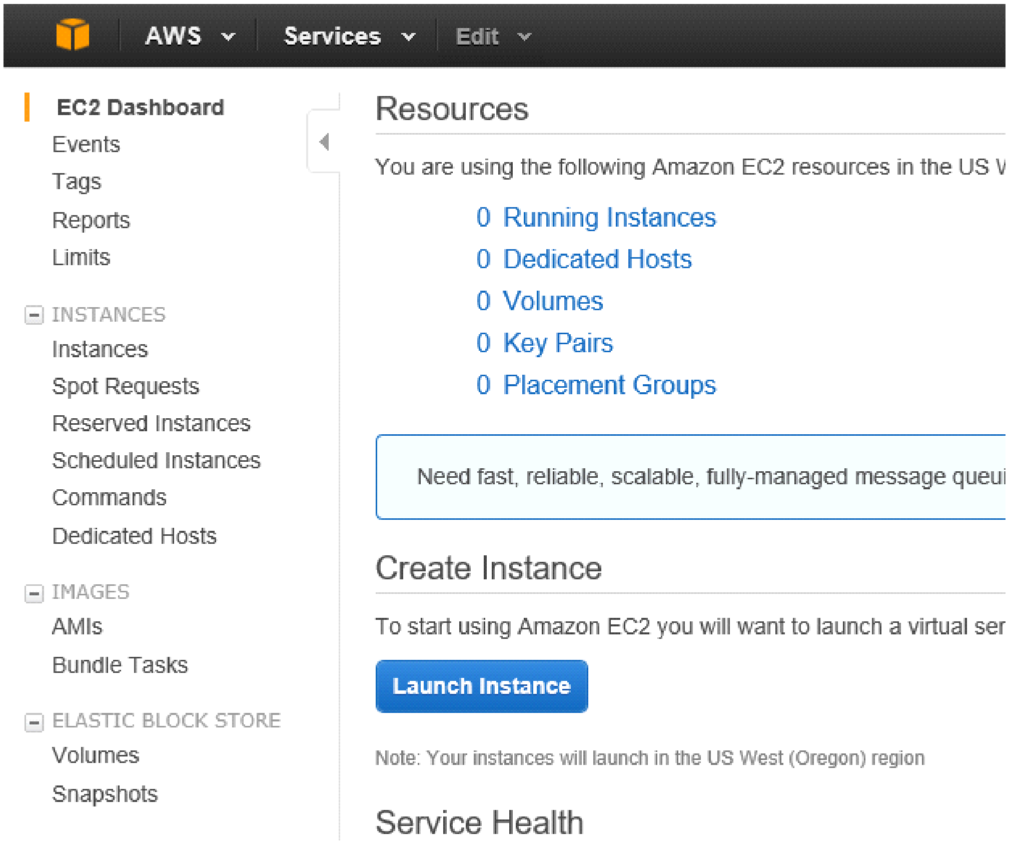
BEGIN

Cloud Formation Network: Here we go…

Step 1: Get an account. Done.

Step 2: Sanitize the account. Done.

Step 3: Launch an EC2 instance. A small one. Like a T2, say.

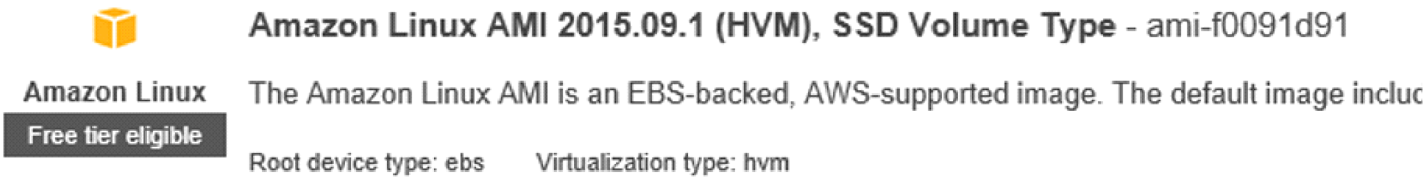


You are using the following Amazon EC2 resources in the US V

Need fast, reliable, scalable, fully-managed message queui

To start using Amazon EC2 you will want to launch a virtual ser

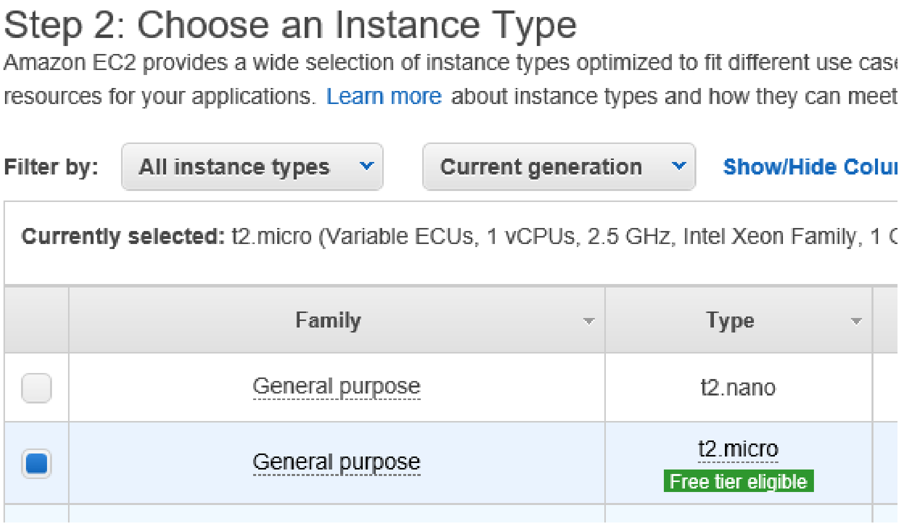
Note: Your instances will launch in the US West (Oregon) region



Amazon Linux AMI 2015.09.1 (HVM), SSD Volume Type - ami-f0091d91

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image incluc

Root device type: ebs Virtualization type: hvm



Step 2: Choose an Instance Type

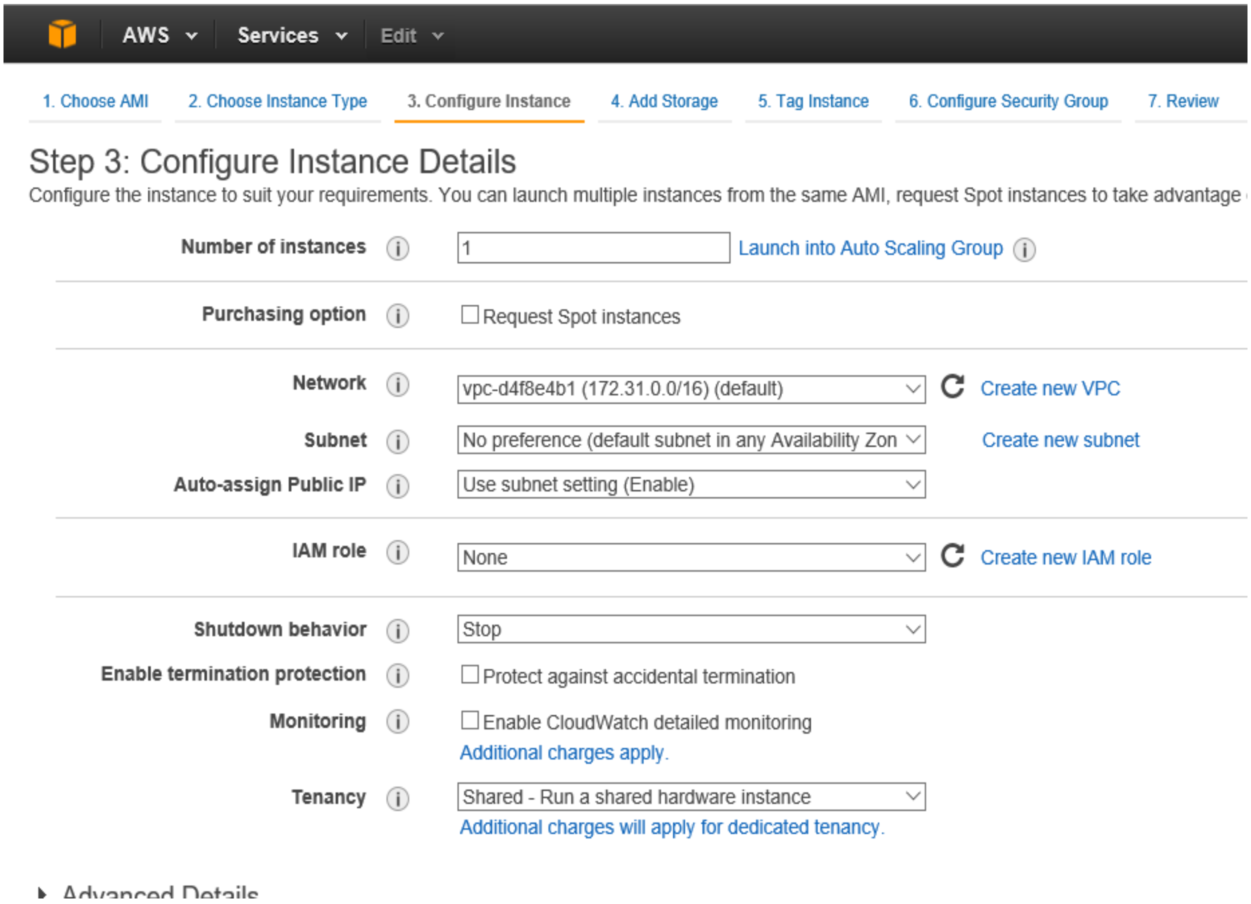
Amazon EC2 provides a wide selection of instance types optimized to fit different use case resources for your applications. Learn more about instance types and how they can meet

Filter by: All instance types Current generation

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 C

	Family	Type
□	General purpose	t2.nano
■	General purpose	t2.micro Free tier eligible

Notice that the t2 micro is selected by default; so Next!



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

Number of instances: 1 Launch into Auto Scaling Group

Purchasing option: □ Request Spot instances

Network: vpc-d4f8e4b1 (172.31.0.0/16) (default) Create new VPC

Subnet: No preference (default subnet in any Availability Zon Create new subnet

Auto-assign Public IP: Use subnet setting (Enable)

IAM role: None Create new IAM role

Shutdown behavior: Stop

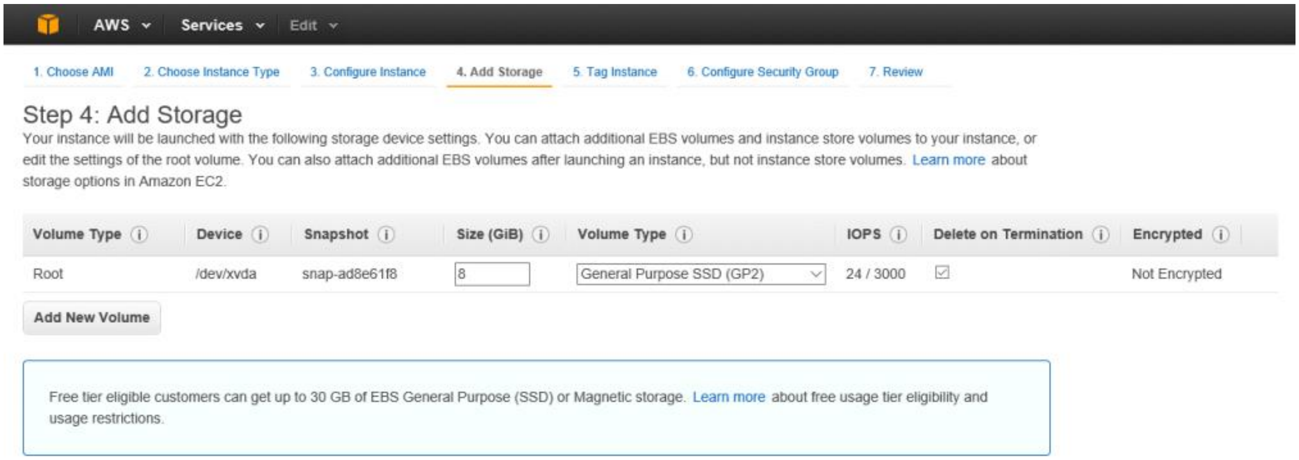
Enable termination protection: □ Protect against accidental termination

Monitoring: □ Enable CloudWatch detailed monitoring
Additional charges apply.

Tenancy: Shared - Run a shared hardware instance
Additional charges will apply for dedicated tenancy.

Advanced Details

Notice that again by default these are just fine so Next!!!



Step 4: Add Storage

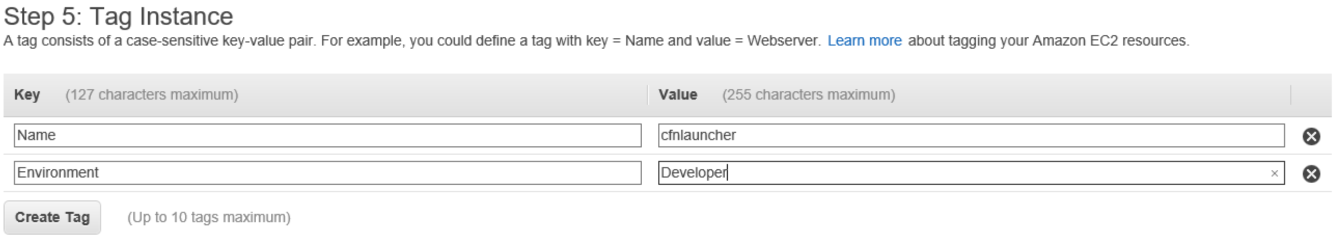
Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes.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

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Delete on Termination	Encrypted
Root	/dev/xvda	snap-ad8e61f8	8	General Purpose SSD (GP2)	24 / 3000	☑	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.

Again: Ok! Next!!!



Step 5: Tag Instance

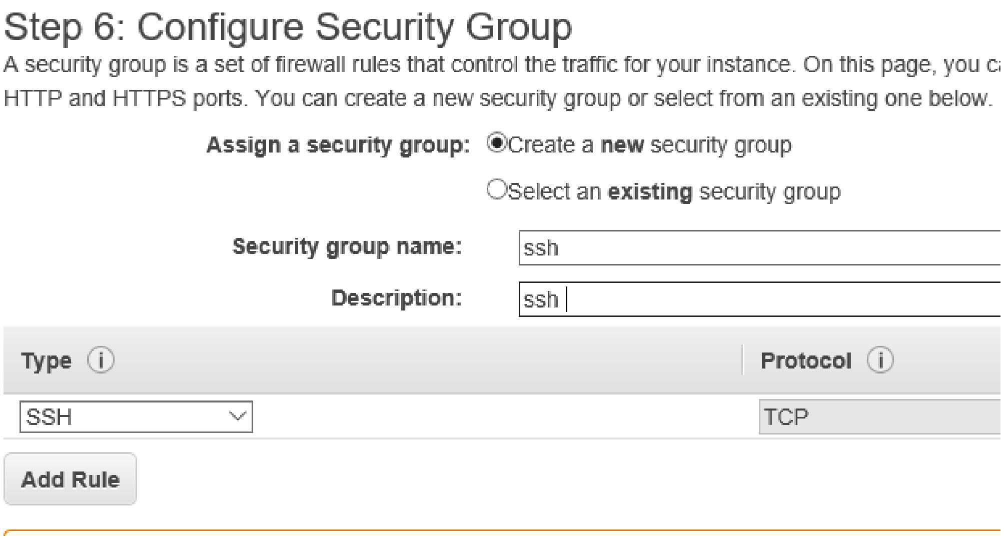
A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. Learn more about tagging your Amazon EC2 resources.

Key (127 characters maximum)	Value (255 characters maximum)
Name	cfnlauncher
Environment	Developer

Create Tag (Up to 10 tags maximum)

Here the Name is a default key; so give a good name like 'cfnlauncher'. Notice I added an Environment also; so this is for my Dev team to work on. Onward!!!

Next we need a security group which you can think of as a firewall around this set of resources. I gave this the following:



Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you c HTTP and HTTPS ports. You can create a new security group or select from an existing one below.

Assign a security group: ● Create a **new** security group

○ Select an **existing** security group

Security group name: ssh

Description: ssh

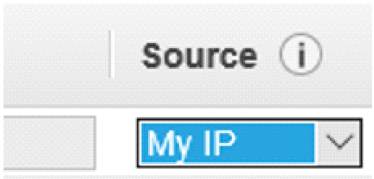
Type	Protocol
SSH	TCP

Add Rule

Kinda dull but accurate…

But there is another important thing to do on this page: Restrict access based on ip…

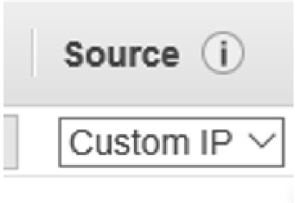
First get my ip:



Source

My IP

The address appears to the right of the blue box (redacted here). You use this and the dropdown to set restrictions on access.



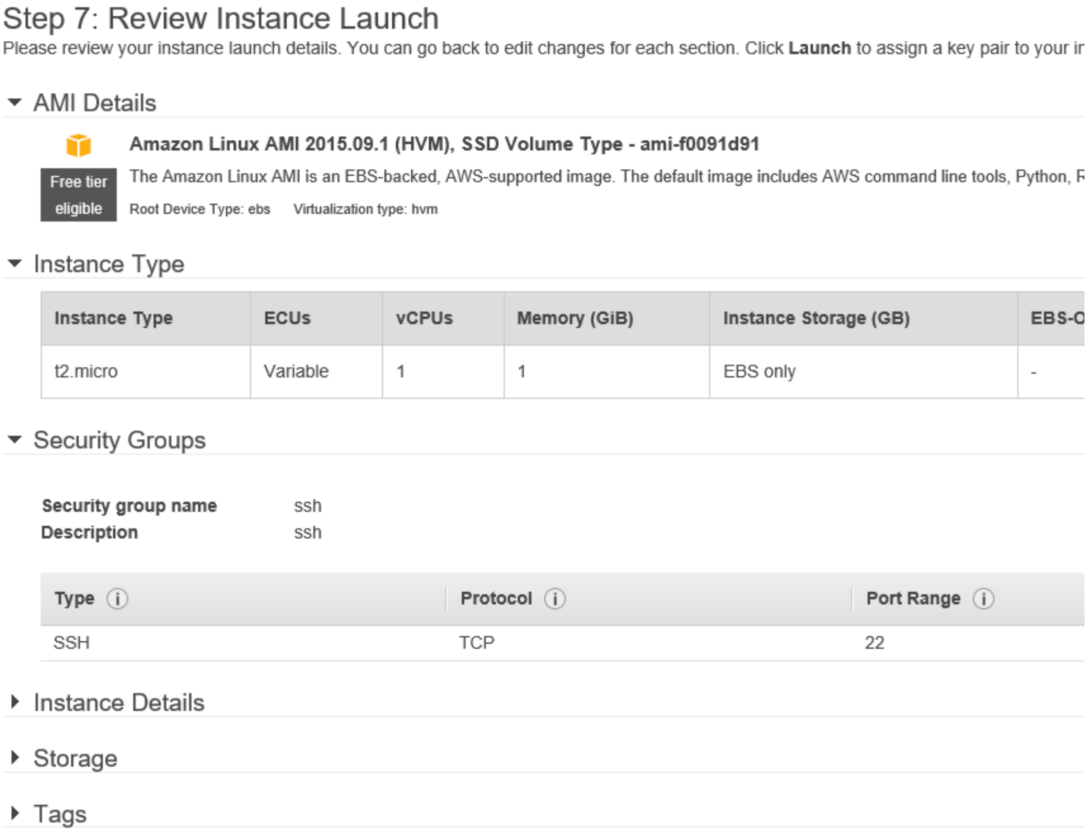
Source

Custom IP

Here you'll have to look up how restrictive / unrestrictive you'd like to be in your approach.

/16 is the least restrictive; so I wound up with something like 121.73.0.0/16 (but not that).

And Review and Launch…



Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your ir

AMI Details

Amazon Linux AMI 2015.09.1 (HVM), SSD Volume Type - ami-f0091d91

Free tier eligible

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, F

Root Device Type: ebs Virtualization type: hvm

Instance Type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-O
t2.micro	Variable	1	1	EBS only	-

Security Groups

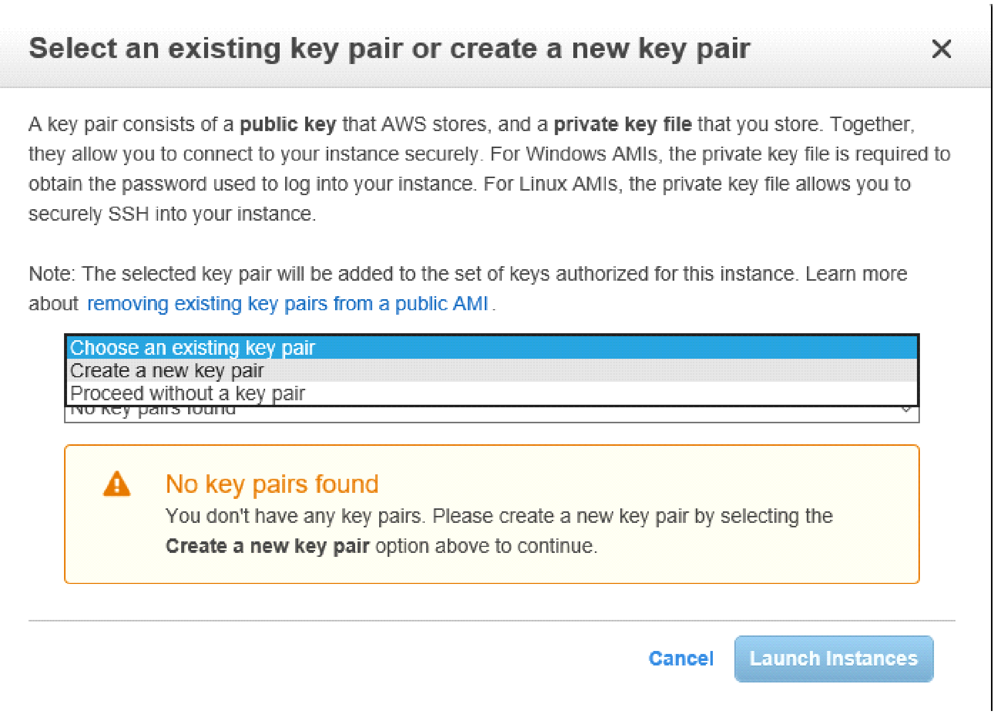
Security group name ssh
Description ssh

Type	Protocol	Port Range
SSH	TCP	22

Instance Details

Storage

Tags



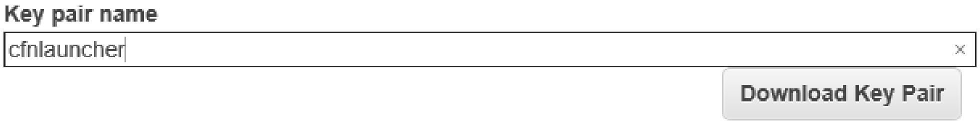
Select an existing key pair or create a new key pair

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 .

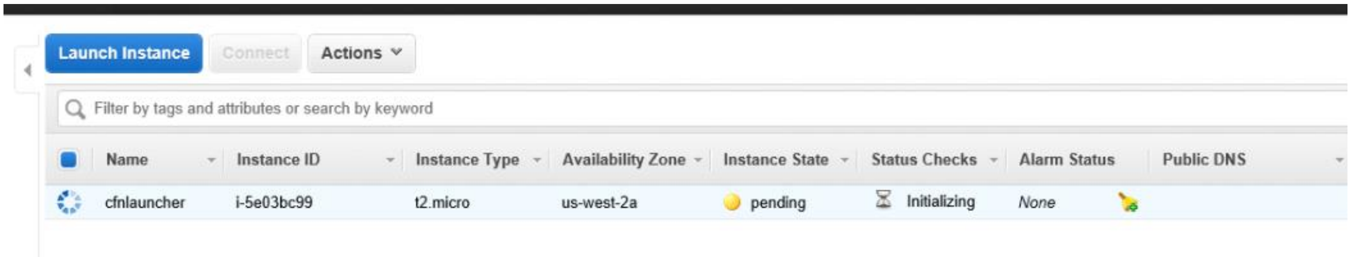
No key pairs found

You don't have any key pairs. Please create a new key pair by selecting the **Create a new key pair** option above to continue.



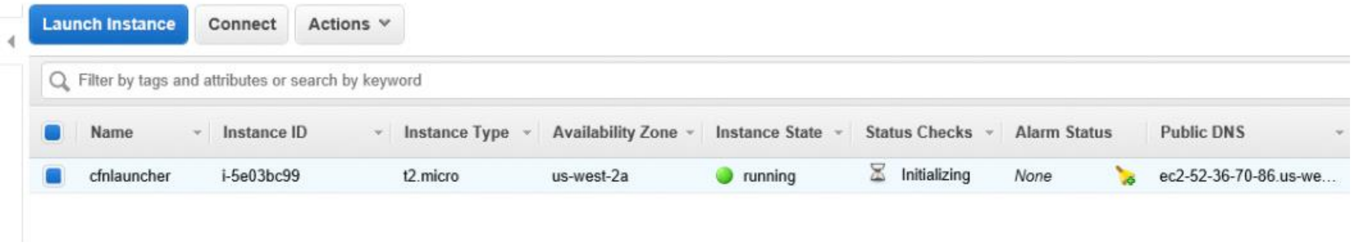
Key pair name: cfnlauncher

So that will download; then Launch Instance button… and click on View Instances blue button lower right that shows up next.



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
cfnlauncher	i-5e03bc99	t2.micro	us-west-2a	pending	Initializing	None	

This becomes, eventually:

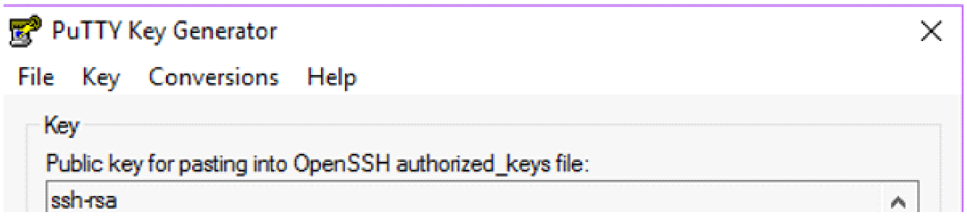


Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
cfnlauncher	i-5e03bc99	t2.micro	us-west-2a	running	Initializing	None	ec2-52-36-70-86.us-we…

Now… download Putty and PuttyGen (in my case for a Windows machine) in order to work with that private key file that we downloaded a moment ago.

Launch PuttyGen.

After-the-fact screencap; notice Load button to Load an existing private key file:

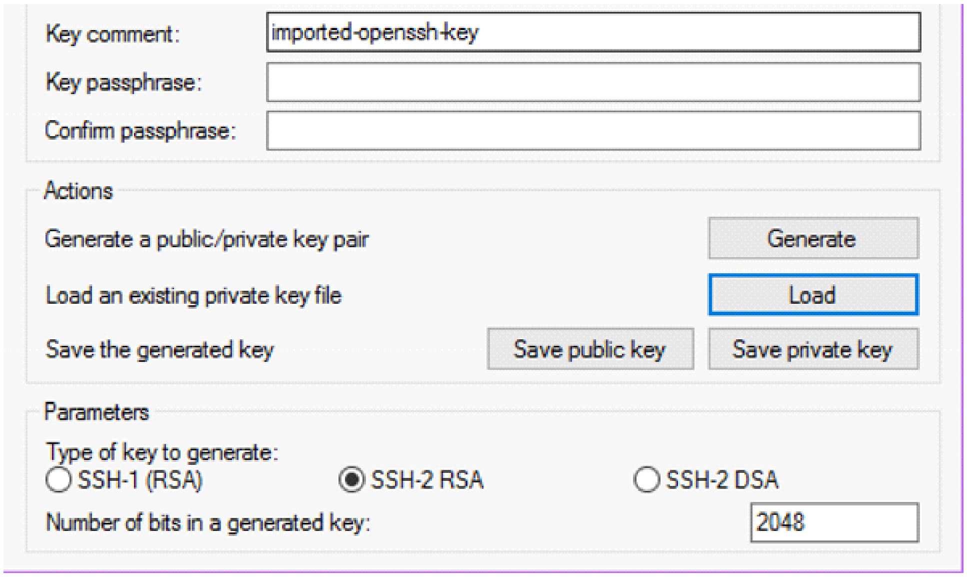


Key

Public key for pasting into OpenSSH authorized_keys file:

ssh-rsa

(redacted information)



Key comment: imported-openssh-key

Key passphrase:

Confirm passphrase:

Actions

Generate a public/private key pair

Load an existing private key file

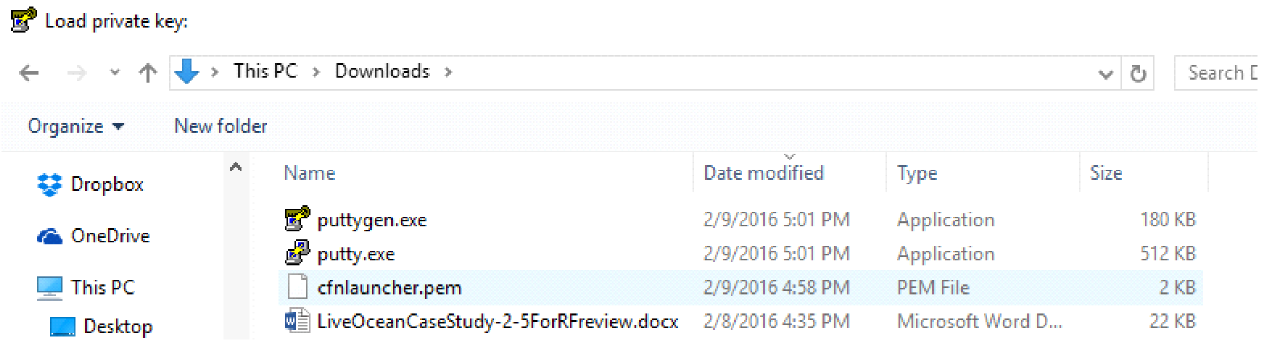
Save the generated key

Parameters

Type of key to generate:

○ SSH-1 (RSA) ● SSH-2 RSA ○ SSH-2 DSA

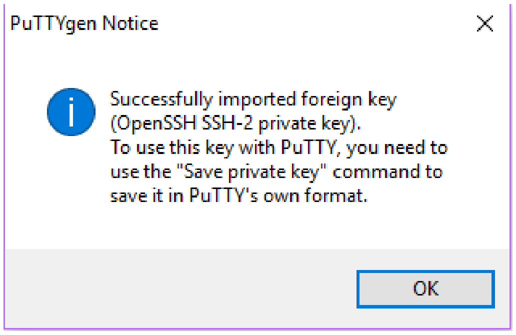
Number of bits in a generated key: 2048



Load private key:

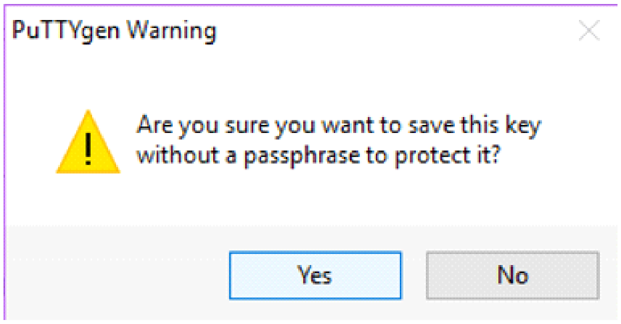
This PC > Downloads >

Name	Date modified	Type	Size
puttygen.exe	2/9/2016 5:01 PM	Application	180 KB
putty.exe	2/9/2016 5:01 PM	Application	512 KB
cfnlauncher.pem	2/9/2016 4:58 PM	PEM File	2 KB
LiveOceanCaseStudy-2-5ForRFreview.docx	2/8/2016 4:35 PM	Microsoft Word D...	22 KB



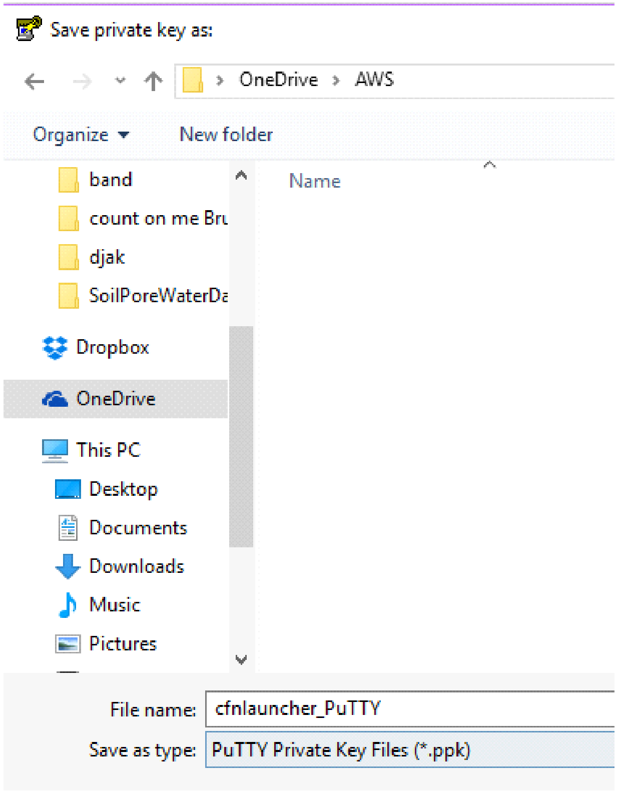
PuTTYgen Notice

Successfully imported foreign key (OpenSSH SSH-2 private key).
To use this key with PuTTY, you need to use the "Save private key" command to save it in PuTTY's own format.



PuTTYgen Warning

Are you sure you want to save this key without a passphrase to protect it?



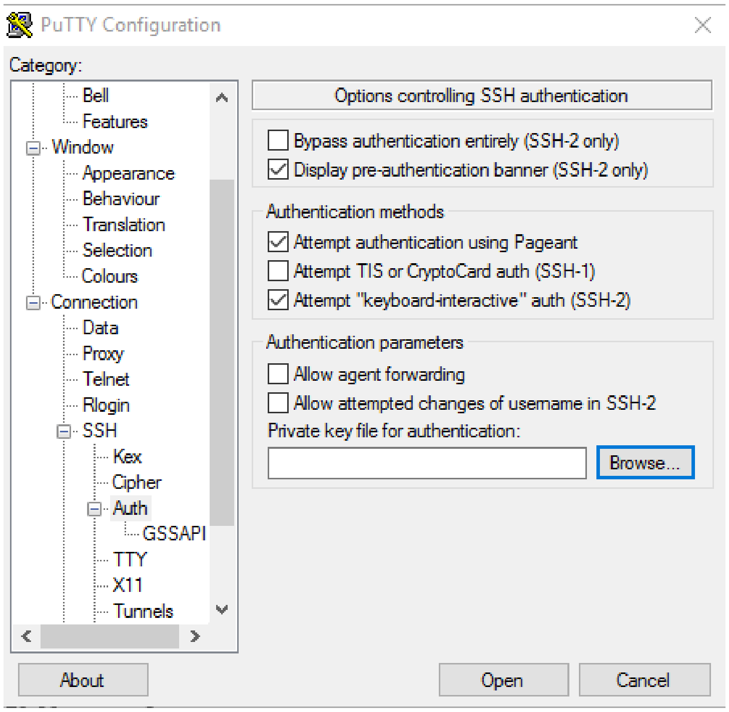
Save private key as:

File name: cfnlauncher_PuTTY

Save as type: PuTTY Private Key Files (*.ppk)

So the new private key file (putty-ized) is saved on my private OneDrive.

Now run PuTTY; but go back to the EC2 instance console to get the IP address:



Options controlling SSH authentication

□ Bypass authentication entirely (SSH-2 only)
☑ Display pre-authentication banner (SSH-2 only)

Authentication methods

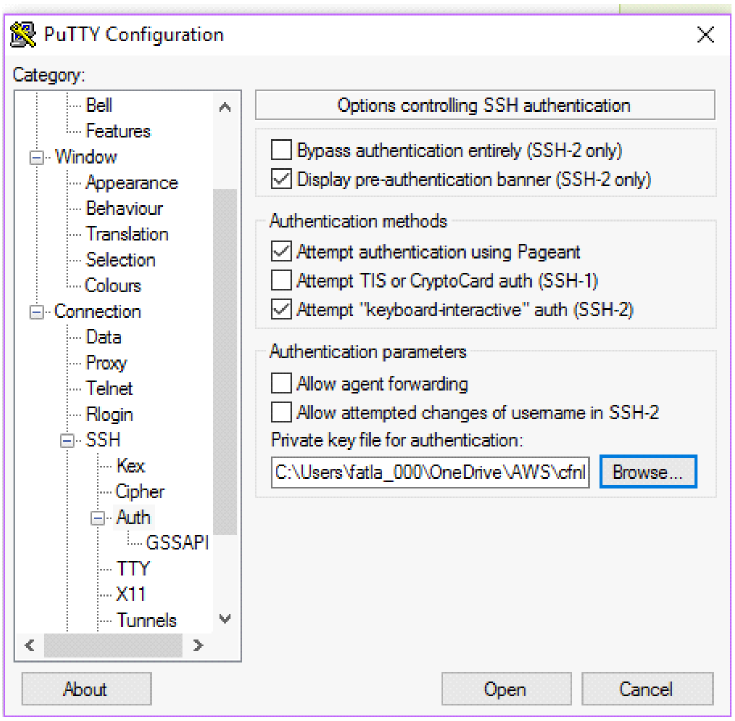
☑ Attempt authentication using Pageant
□ Attempt TIS or CryptoCard auth (SSH-1)
☑ Attempt "keyboard-interactive" auth (SSH-2)

Authentication parameters

□ Allow agent forwarding
□ Allow attempted changes of username in SSH-2

Private key file for authentication: [Browse...]

Browse to that new private key file…



Options controlling SSH authentication

□ Bypass authentication entirely (SSH-2 only)
☑ Display pre-authentication banner (SSH-2 only)

Authentication methods

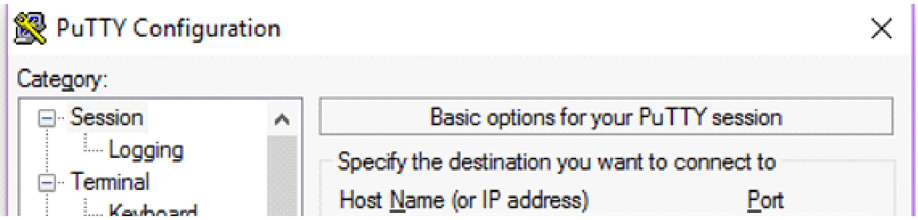
☑ Attempt authentication using Pageant
□ Attempt TIS or CryptoCard auth (SSH-1)
☑ Attempt "keyboard-interactive" auth (SSH-2)

Authentication parameters

□ Allow agent forwarding
□ Allow attempted changes of username in SSH-2

Private key file for authentication: C:\Users\fatla_000\OneDrive\AWS\cfnl [Browse...]

Now let's save this as a PuTTY session to make it easier to do next time.
But if I ever shut down this instance and bring it back up you get a new IP address so this will be moot.

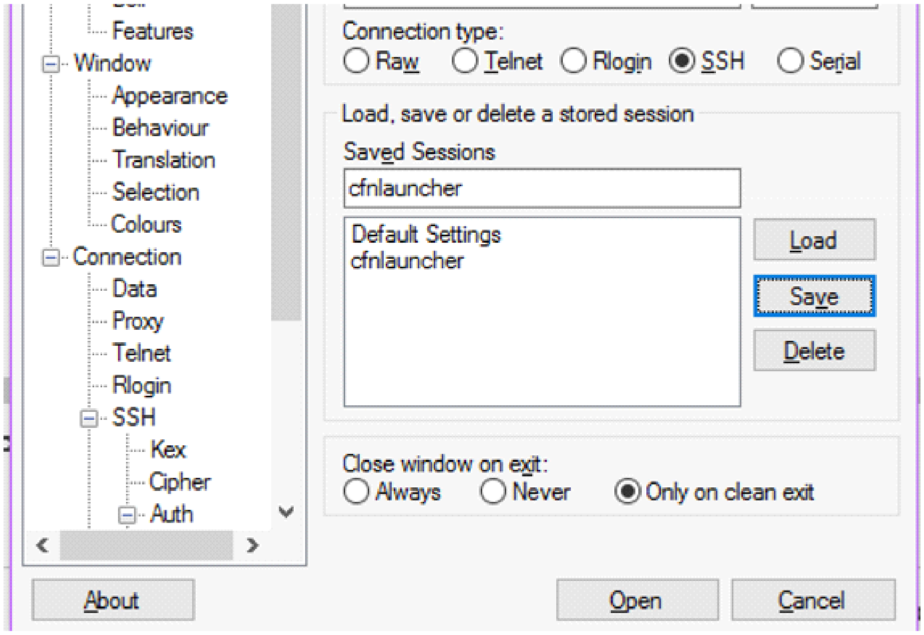


Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address) Port

Redacted IP address…



Connection type:
○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

Load, save or delete a stored session

Saved Sessions

cfnlauncher

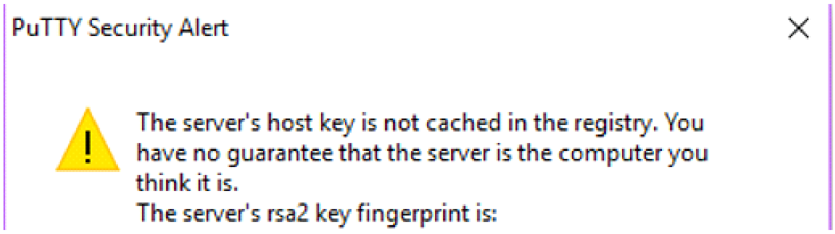
Default Settings
cfnlauncher

Close window on exit:
○ Always ○ Never ● Only on clean exit

Please notice two things:

1. The Host Name begins with ec2-user@ then the ip.
2. Save the Session before you click the Open button.

And one time we will get this warning:

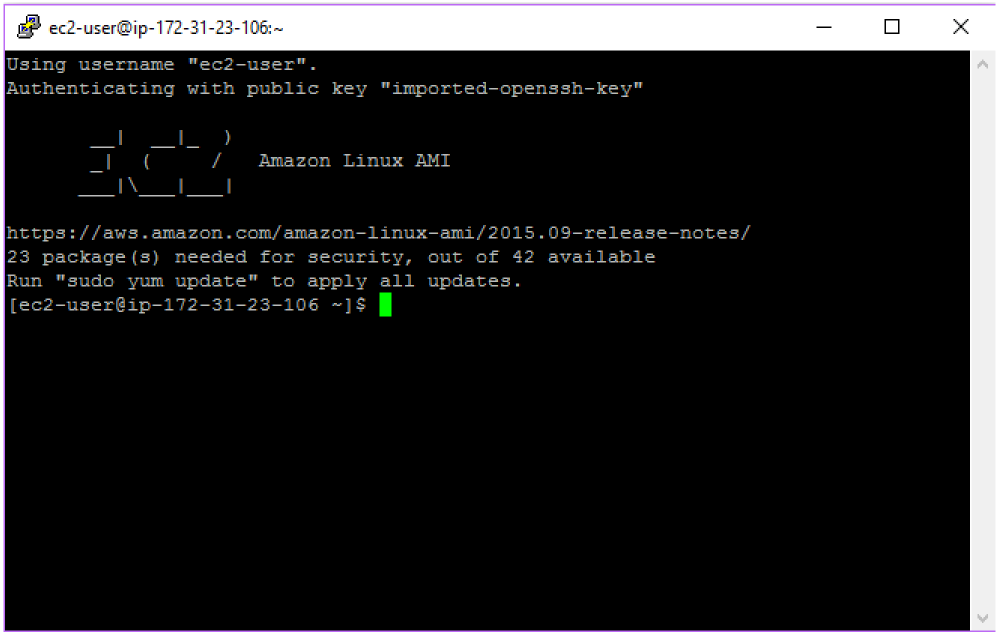


PuTTY Security Alert

The server's host key is not cached in the registry. You have no guarantee that the server is the computer you think it is.
The server's rsa2 key fingerprint is:

Etcetera redacted; it is asking if you want to continue. The answer is Yes.

Now (TaDAAAAA) here is your console:

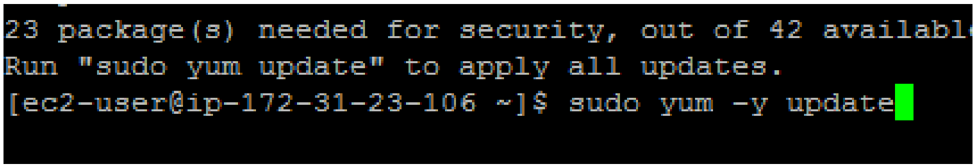


```
Using username "ec2-user".
Authenticating with public key "imported-openssh-key"

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

https://aws.amazon.com/amazon-linux-ami/2015.09-release-notes/
23 package(s) needed for security, out of 42 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-23-106 ~]$
```

Now let's install all the most recent patches/upgrades…

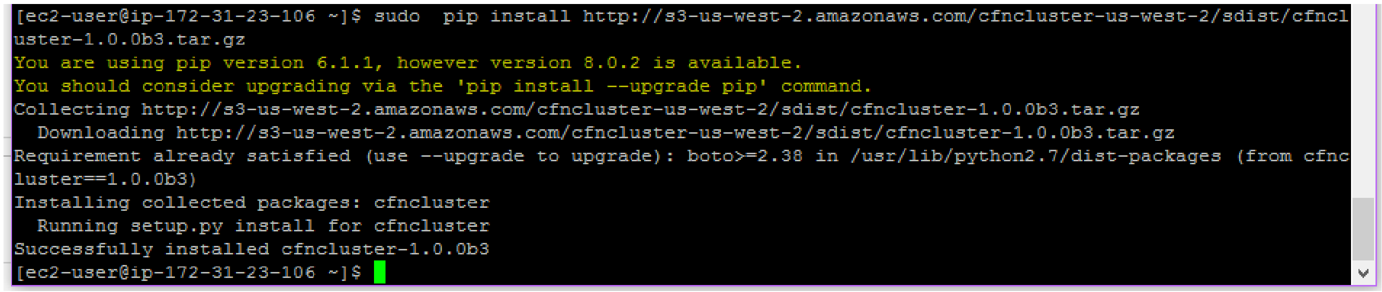


```
23 package(s) needed for security, out of 42 availabl
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-23-106 ~]$ sudo yum -y update
```

Now here is the cfn install command; this is CFN Beta and it is fine to use:

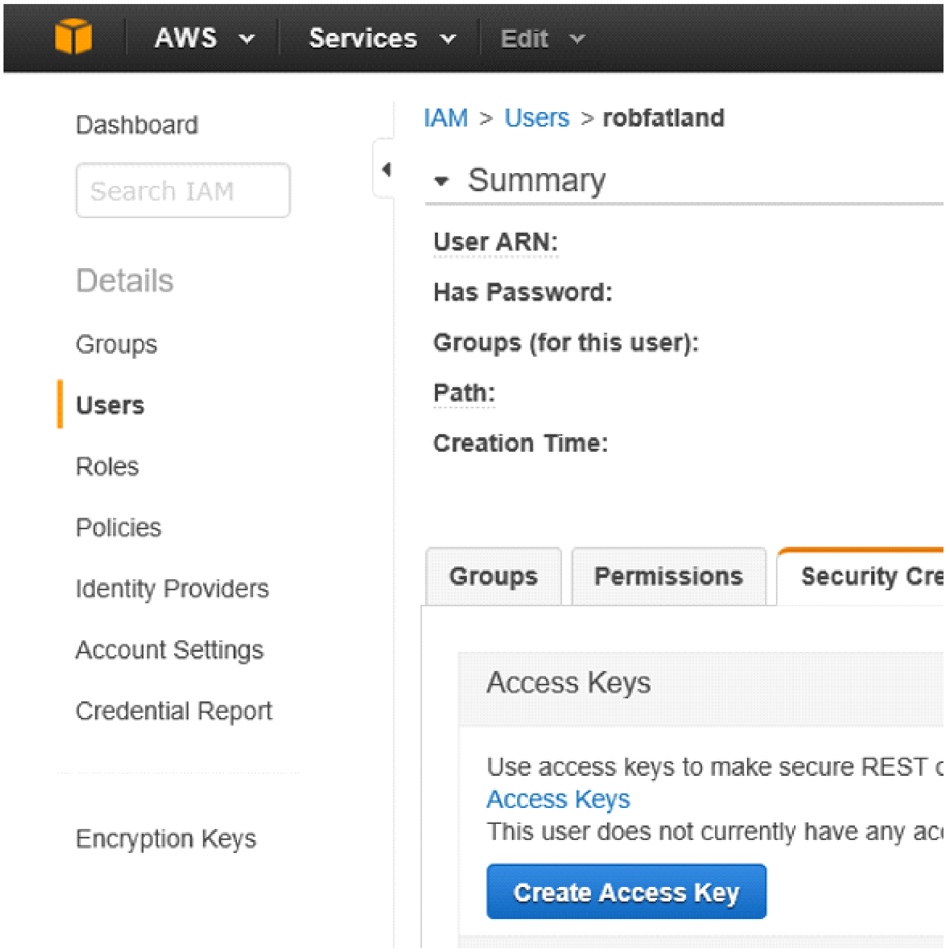
sudo pip install http://s3-us-west-2.amazonaws.com/cfncluster-us-west-2/sdist/cfncluster-1.0.0b3.tar.gz

(should be plain text really)



```
[ec2-user@ip-172-31-23-106 ~]$ sudo  pip install http://s3-us-west-2.amazonaws.com/cfncluster-us-west-2/sdist/cfncl
uster-1.0.0b3.tar.gz
You are using pip version 6.1.1, however version 8.0.2 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
Collecting http://s3-us-west-2.amazonaws.com/cfncluster-us-west-2/sdist/cfncluster-1.0.0b3.tar.gz
  Downloading http://s3-us-west-2.amazonaws.com/cfncluster-us-west-2/sdist/cfncluster-1.0.0b3.tar.gz
Requirement already satisfied (use --upgrade to upgrade): boto>=2.38 in /usr/lib/python2.7/dist-packages (from cfnc
luster==1.0.0b3)
Installing collected packages: cfncluster
  Running setup.py install for cfncluster
Successfully installed cfncluster-1.0.0b3
[ec2-user@ip-172-31-23-106 ~]$
```

And now DANGER: We are getting into access keys so be careful. Go to IAM and click on the User (me) to grant access to…



User ARN:

Has Password:

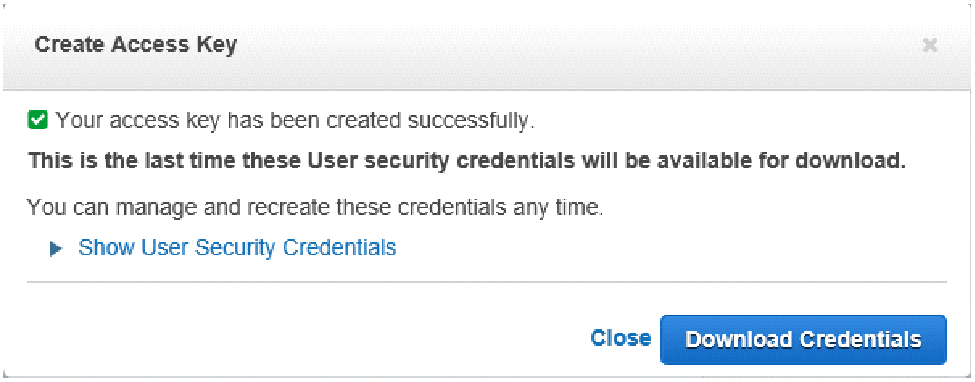
Groups (for this user):

Path:

Creation Time:

Access Keys

Use access keys to make secure REST c
Access Keys
This user does not currently have any ac



☑ Your access key has been created successfully.

This is the last time these User security credentials will be available for download.

You can manage and recreate these credentials any time.

Run on the command line: cfncluster configure. I do not include screencaps because key information is involved. But there is a URL to follow:

You can use that as a guide.

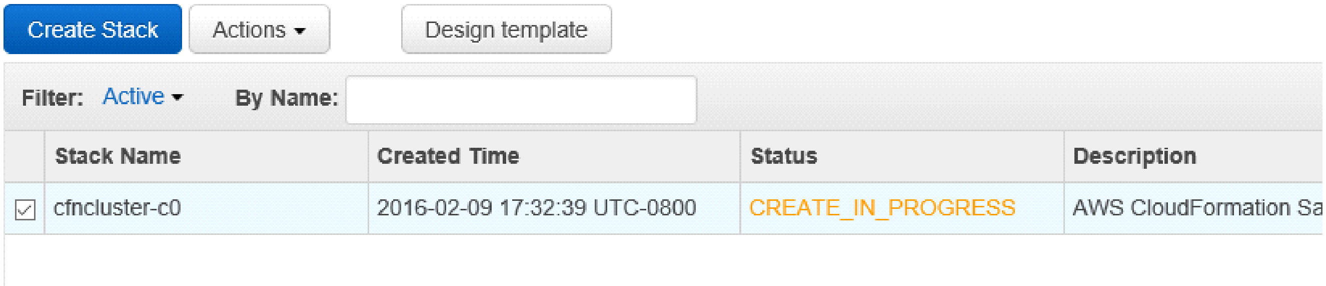
Now run

cfncluster create c0

This will create a cluster called 'c0' including a head node. I will be paying for this head node until I turn it off. The default is a t2 (so small)… go to the AWS console to see it.

Cloud Formation gives you a sorta real-time picture of how it is coming together.

Keep hitting refresh if you are impatient

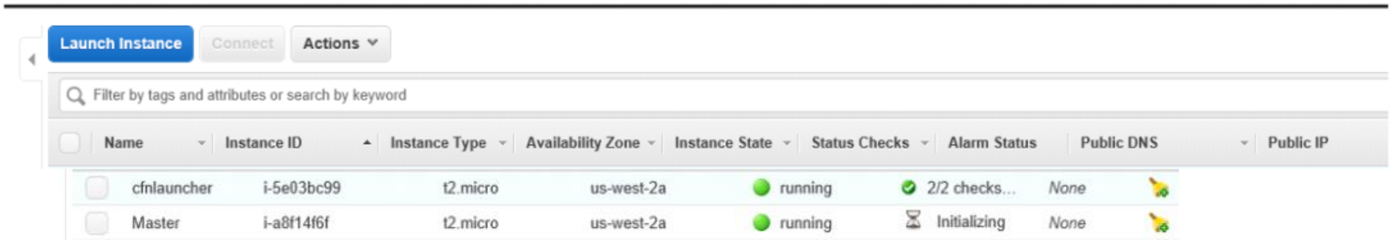


Filter: Active ▾ By Name:

	Stack Name	Created Time	Status	Description
☑	cfncluster-c0	2016-02-09 17:32:39 UTC-0800	CREATE_IN_PROGRESS	AWS CloudFormation Sa

Use Edit dropdown in the top toolbar to drag an icon to the bar to make it easier to do; I did EC2.

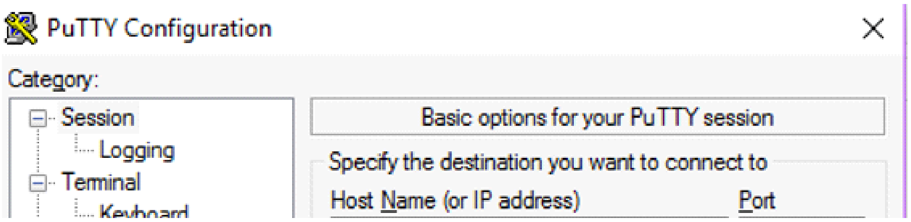
Once it completes… now we have a Head Node.



	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS	Public IP
□	cfnlauncher	i-5e03bc99	t2.micro	us-west-2a	● running	✓ 2/2 checks...	None		
□	Master	i-a8f14f6f	t2.micro	us-west-2a	● running	⌛ Initializing	None		

Now coming back to this after some coffee… how do I log in to my Master node??

Well let's launch PuTTY

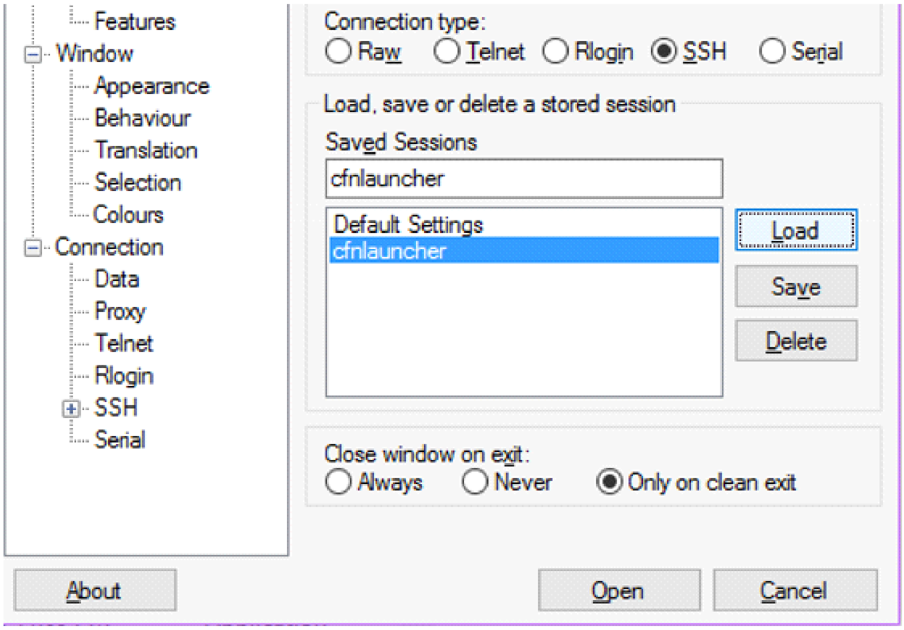


Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address) Port

Redacted…



Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

Load, save or delete a stored session

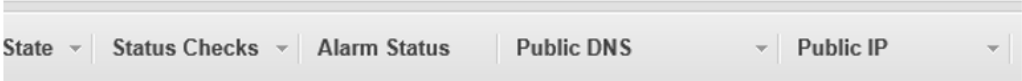
Saved Sessions: cfnlauncher

Default Settings
cfnlauncher

Close window on exit: ○ Always ○ Never ● Only on clean exit

And select cfnlauncher and then we will customize that to our ip address for the Master node.

Get the IP address from the AWS Console (and we get that warning en route)

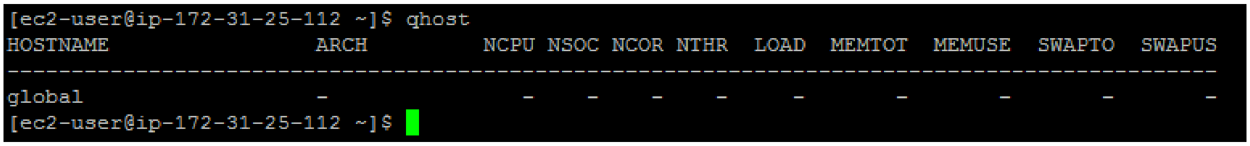


State	Status Checks	Alarm Status	Public DNS	Public IP

Here we are; and we can be root without knowing the password… sudo su -

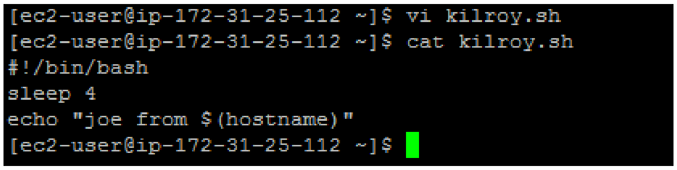
We could also do sudo adduser to add people if we like.

I am 'global' the head node and there are no workers; so say qhost to see this:



```
[ec2-user@ip-172-31-25-112 ~]$ qhost
HOSTNAME                ARCH         NCPU NSOC NCOR NTHR  LOAD  MEMTOT  MEMUSE  SWAPTO  SWAPUS
----------------------------------------------------------------------------------------------
global                  -               -    -    -    -     -       -       -       -       -
[ec2-user@ip-172-31-25-112 ~]$
```

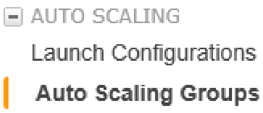
Now let's create a script to do nothing in particular:



```
[ec2-user@ip-172-31-25-112 ~]$ vi kilroy.sh
[ec2-user@ip-172-31-25-112 ~]$ cat kilroy.sh
#!/bin/bash
sleep 4
echo "joe from $(hostname)"
[ec2-user@ip-172-31-25-112 ~]$
```

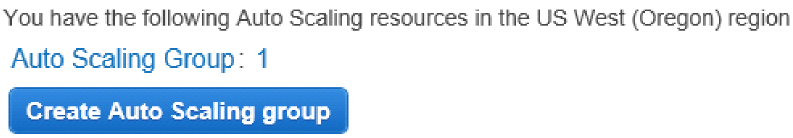
Now let's make kilroy.sh executable with chmod and then submit it to the queue via 'qsub kilroy.sh'…

This failed (although the jobs are on the queue) so we return to the console and go after



In the left sidebar.

And now in here:

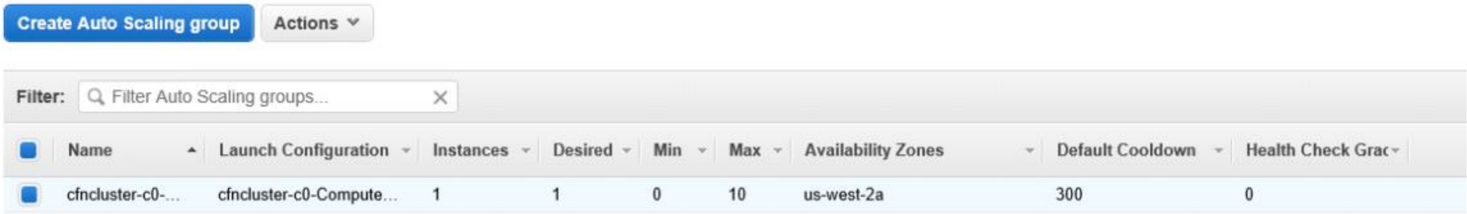


You have the following Auto Scaling resources in the US West (Oregon) region

Auto Scaling Group : 1

We will click on 'Auto Scaling Group'

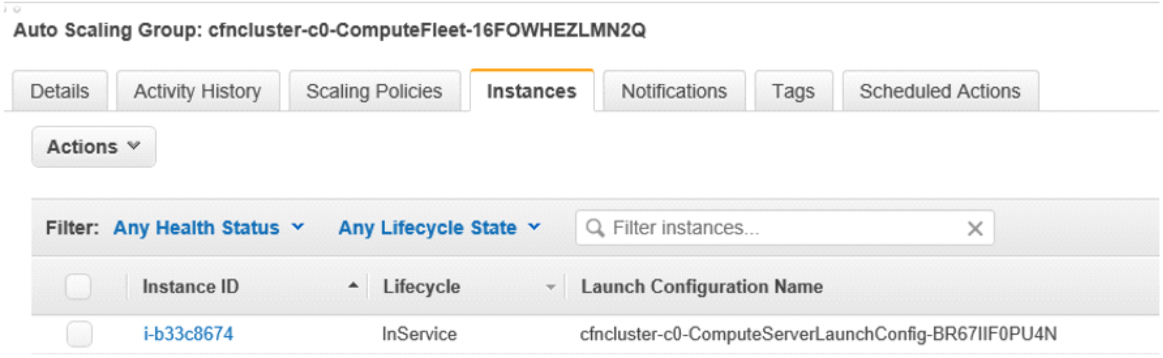
And here we are…



Name	Launch Configuration	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health Check Grac
cfncluster-c0-...	cfncluster-c0-Compute...	1	1	0	10	us-west-2a	300	0

And at the bottom of this page is a second panel… make that larger using the icons at lower right:

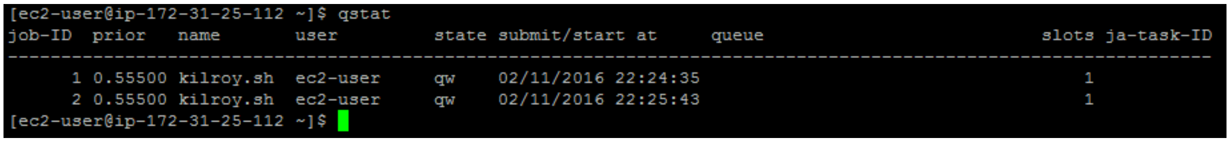
And here we see what that lower pane knows about our Worker starting up:



Instance ID	Lifecycle	Launch Configuration Name
i-b33c8674	InService	cfncluster-c0-ComputeServerLaunchConfig-BR67IIF0PU4N

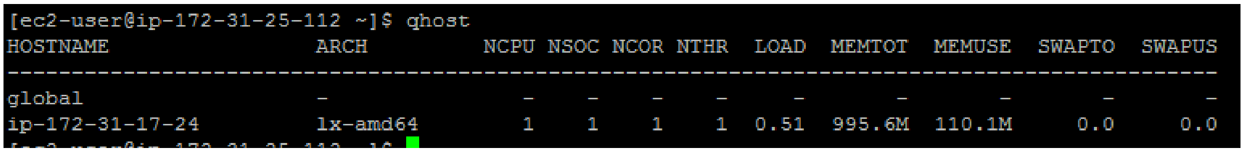
So this is in progress; and as the machine spins up it will eventually show up in 'qhost' on the head node...

This seems to take a few minutes. I ran kilroy twice using qsub so here is the queue using qstat: Both jobs are still present and have no way of running until a Worker actually appears.



```
[ec2-user@ip-172-31-25-112 ~]$ qstat
job-ID  prior   name       user         state submit/start at     queue                          slots ja-task-ID
-----------------------------------------------------------------------------------------------------------------
      1 0.55500 kilroy.sh  ec2-user     qw    02/11/2016 22:24:35                                    1
      2 0.55500 kilroy.sh  ec2-user     qw    02/11/2016 22:25:43                                    1
[ec2-user@ip-172-31-25-112 ~]$
```

However qhost shows that nothing is there 'registered' as part of the cluster yet. Until it did (estimate 3 minutes maybe)

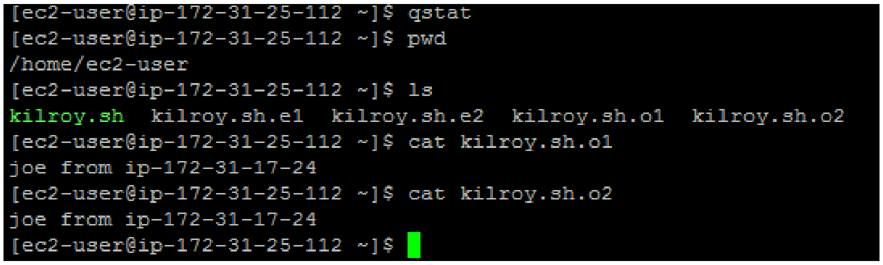


```
[ec2-user@ip-172-31-25-112 ~]$ qhost
HOSTNAME                ARCH         NCPU NSOC NCOR NTHR  LOAD  MEMTOT  MEMUSE  SWAPTO  SWAPUS
----------------------------------------------------------------------------------------------
global                  -               -    -    -    -     -       -       -       -       -
ip-172-31-17-24         lx-amd64        1    1    1    1  0.51  995.6M  110.1M     0.0     0.0
[ec2-user@ip-172-31-25-112 ~]$
```

And now my kilroys ran to completion so qstat shows an empty queue.

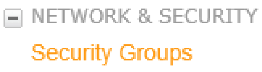
That Worker will stay up for about 55 minutes since I am billed by the hour. Then it will evaporate if it is not doing anything. So I have 54 more minutes to run single-node-cluster experiments... if I want to. Let's run kilroy again. Where is the output going??? It goes to the home directory on the Head Node. So let's go there and check it out.

Here you have it, stdout and stderr:

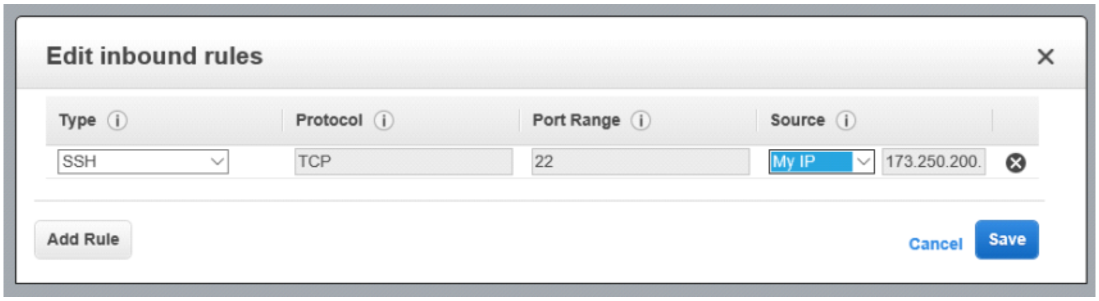


```
[ec2-user@ip-172-31-25-112 ~]$ qstat
[ec2-user@ip-172-31-25-112 ~]$ pwd
/home/ec2-user
[ec2-user@ip-172-31-25-112 ~]$ ls
kilroy.sh  kilroy.sh.e1  kilroy.sh.e2  kilroy.sh.o1  kilroy.sh.o2
[ec2-user@ip-172-31-25-112 ~]$ cat kilroy.sh.o1
joe from ip-172-31-17-24
[ec2-user@ip-172-31-25-112 ~]$ cat kilroy.sh.o2
joe from ip-172-31-17-24
[ec2-user@ip-172-31-25-112 ~]$
```

How is the configuration working? It is split between the initiation node and the head node. Let's log back into cfncluster (what I call the initiation node) to see where the first part of that lives. I tried to do this with PuTTY but it failed even though I was loading a stored profile. The problem has to do with security... Look at the menu on the left:



Click on Edit; oh dear my ip address has changed…


Edit inbound rules

Type	Protocol	Port Range	Source	
SSH	TCP	22	My IP	173.250.200.

I simply opened up access to 'anywhere' as I am in a hurry (uh oh) and now I can get back to my machine.

Now that we've come this far the next step is to review the config process.

The documentation for this file is at http://cfncluster.readthedocs.org/en/latest/configuration.html