AWS Parallel Cluster Tutorial

By: Michael DeProspo, Nathaniel Law, Thomas Seeley, and Madison Jones

Outline:

Prerequisites:

This tutorial requires the following prerequisites:

- Python 3 or higher
- AWS account - free tier
    - This will charge your Credit Card $1 to create but it will be refunded as soon as payment has been verified
- AWS CLI installed (https://aws.amazon.com/cli/)
    - If you need to install this instructions can be found here: https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-install.html

Setting up AWS Parallel Cluster:

This section will go over how to install the parallel cluster virtual environment, setup a non-root user account with administrator privileges, and create a key pair so that you can login remotely to the instance nodes.

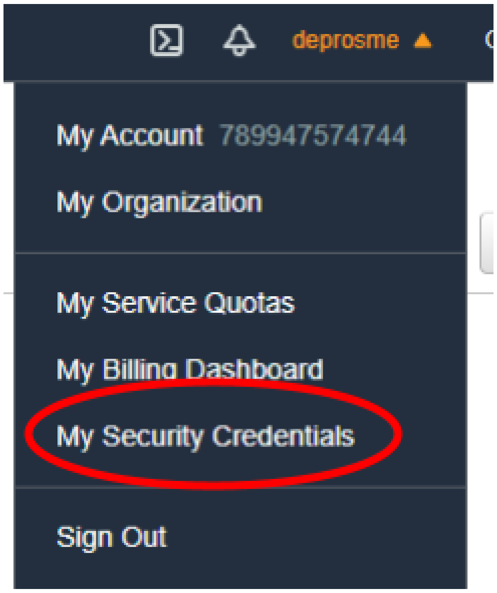
Parallel Cluster Installation:

- Check to make sure python 3 is your python version with:
    - `python --version`
- Install parallel cluster with the following instructions OS specific instructions
    - Windows:
        - `pip3 install --user --upgrade virtualenv`
        - `virtualenv %USERPROFILE%\apc-xe`
        - `%USERPROFILE%\apc-xe\Scripts\activate`
        - `pip3 install --upgrade aws-parallelcluster`
    - Linux/macOS:
        - `python3 -m pip install --upgrade pip`
        - `python3 -m pip install --user --upgrade virtualenv`

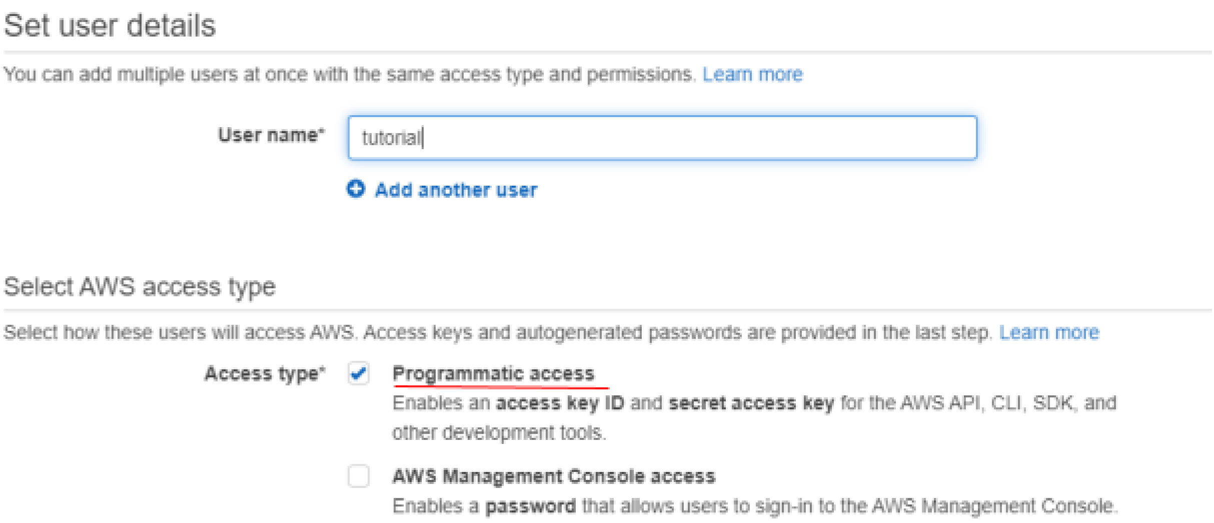
        - `python3 -m virtualenv ~/apc-xe`
        - `source ~/apc-xe/bin/activate`
        - `python3 -m pip install --upgrade aws-parallelcluster`
- Run “`pcluster version`” in the terminal to verify installation was successful (any system)
- Ensure that the version is 2.10.3 or higher
- If it’s lower repeat the previous steps in a new virtual environment
    - If there is an error message that says “command not found” make sure you are in the virtual environment by running
        - MacOS/Linux: `source ~/apc-ve/bin/activate`
        - Windows: `C:\>%USERPROFILE%\apc-ve\Scripts\activate`
    - **Additional elaboration on any of the previous steps is here:** https://docs.aws.amazon.com/parallelcluster/latest/ug/install-virtualenv.html
- When finished in the virtual environment you can run the “`deactivate`” command to exit, however all of the terminal commands for the rest of the tutorial will be done in the virtual environment so you do not need to deactivate yet.

User Account Setup:

- Next go to http://aws.amazon.com/ and login
- Click on your username in top right corner and go to my security credentials



- On the left side of the screen under **access management** click on users and select **Add user**
- The username does not matter, **make sure programmatic access is checked**, console access should not be checked



Set user details

You can add multiple users at once with the same access type and permissions. Learn more

User name* tutorial

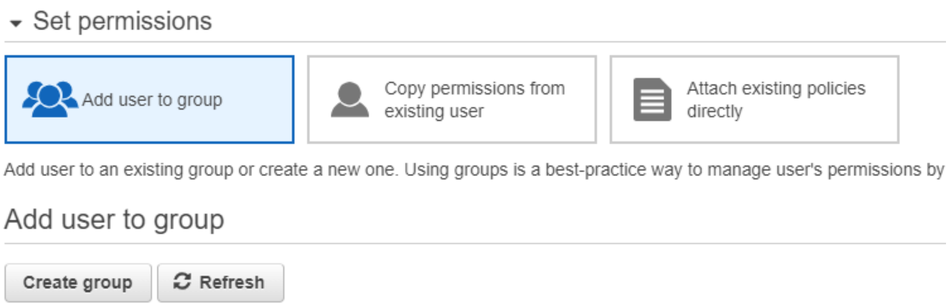
Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. Learn more

Access type* ☑ **Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

□ **AWS Management Console access**
Enables a **password** that allows users to sign-in to the AWS Management Console.

- If there is no group with AdministratorAccess permission already press the create group button



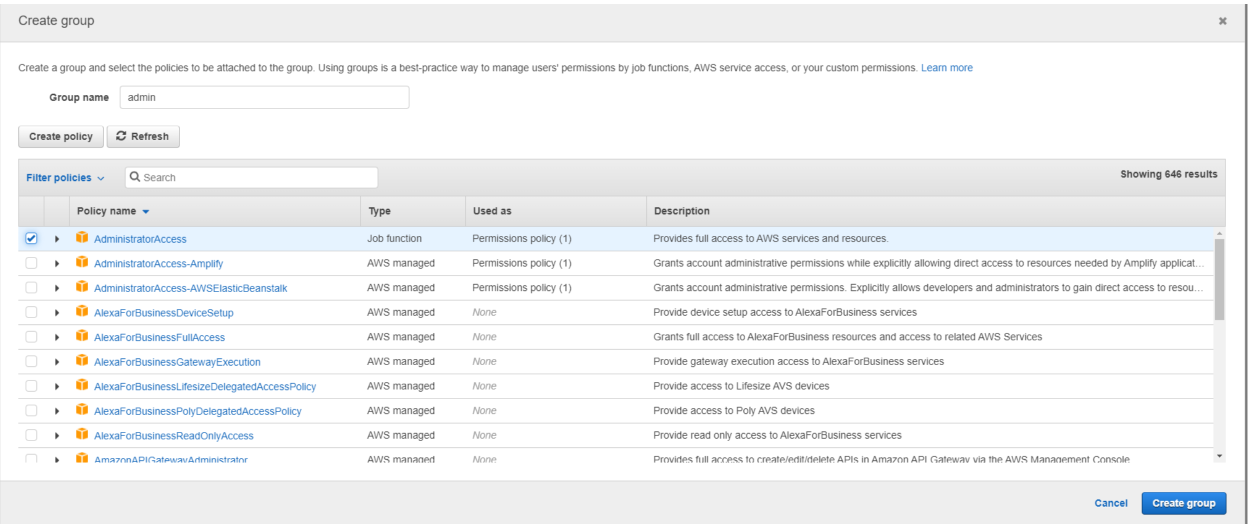
Set permissions

Add user to group
Copy permissions from existing user
Attach existing policies directly

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by

Add user to group

- Create a group policy with the following permissions **don't click create policy, click create group**



Create group

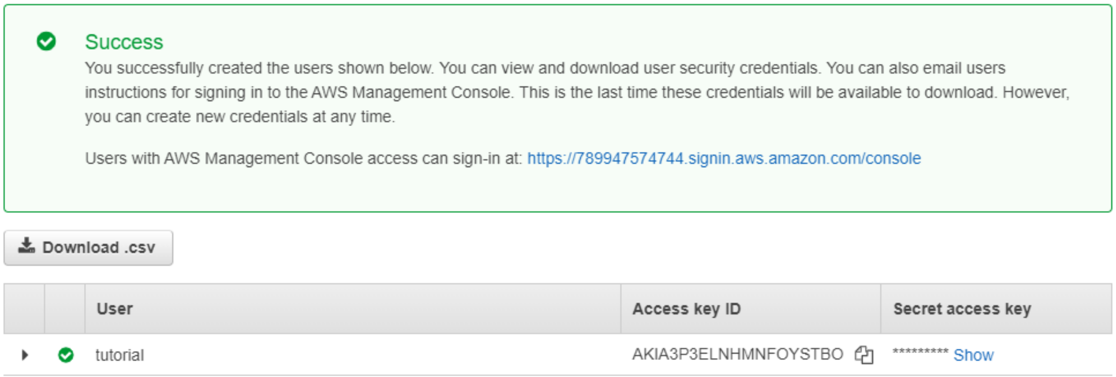
Create a group and select the policies to be attached to the group. Using groups is a best-practice way to manage users' permissions by job functions, AWS service access, or your custom permissions. Learn more

Group name: admin

Showing 646 results

		Policy name	Type	Used as	Description
☑	▸	AdministratorAccess	Job function	Permissions policy (1)	Provides full access to AWS services and resources.
□	▸	AdministratorAccess-Amplify	AWS managed	Permissions policy (1)	Grants account administrative permissions while explicitly allowing direct access to resources needed by Amplify applicat…
□	▸	AdministratorAccess-AWSElasticBeanstalk	AWS managed	Permissions policy (1)	Grants account administrative permissions. Explicitly allows developers and administrators to gain direct access to resou…
□	▸	AlexaForBusinessDeviceSetup	AWS managed	*None*	Provide device setup access to AlexaForBusiness services
□	▸	AlexaForBusinessFullAccess	AWS managed	*None*	Grants full access to AlexaForBusiness resources and access to related AWS Services
□	▸	AlexaForBusinessGatewayExecution	AWS managed	*None*	Provide gateway execution access to AlexaForBusiness services
□	▸	AlexaForBusinessLifesizeDelegatedAccessPolicy	AWS managed	*None*	Provide access to Lifesize AVS devices
□	▸	AlexaForBusinessPolyDelegatedAccessPolicy	AWS managed	*None*	Provide access to Poly AVS devices
□	▸	AlexaForBusinessReadOnlyAccess	AWS managed	*None*	Provide read only access to AlexaForBusiness services
□	▸	AmazonAPIGatewayAdministrator	AWS managed	*None*	Provides full access to create/edit/delete APIs in Amazon API Gateway via the AWS Management Console

- Add your user to this group by hitting next
- Don’t enter any tags unless you desire, hit next
- Click **create user**
- Click download csv
    - **IMPORTANT:** Download this csv file after creating the user and make sure you can find it, this will be the only place to find the secret key ID



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.

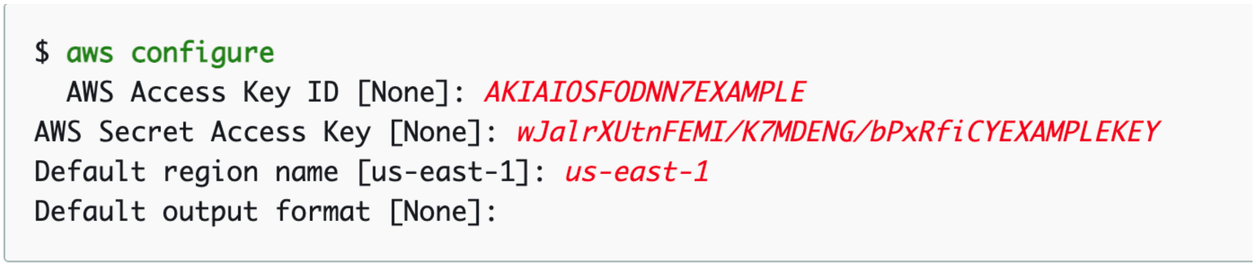
Users with AWS Management Console access can sign-in at: https://789947574744.signin.aws.amazon.com/console

		User	Access key ID	Secret access key
▸	✓	tutorial	AKIA3P3ELNHMNFOYSTBO	********* Show

AWS Configuration:

- From the command line in the virtual environment run the command “`aws configure`”
- First type in your Access Key ID from the csv file that was downloaded and hit enter
- Next type in your Secret Access Key from the csv file and hit enter
- Type in us-east-1 and hit enter
- Type in json and hit enter

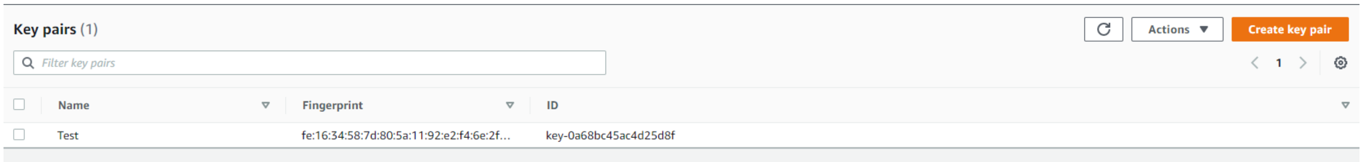
The prompts should look like this:



```
$ aws configure
  AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [us-east-1]: us-east-1
Default output format [None]:
```

Key Pair Creation:

- Go to this link: https://console.aws.amazon.com/ec2/v2/home#KeyPairs
- Click create keypair in upper righthand corner

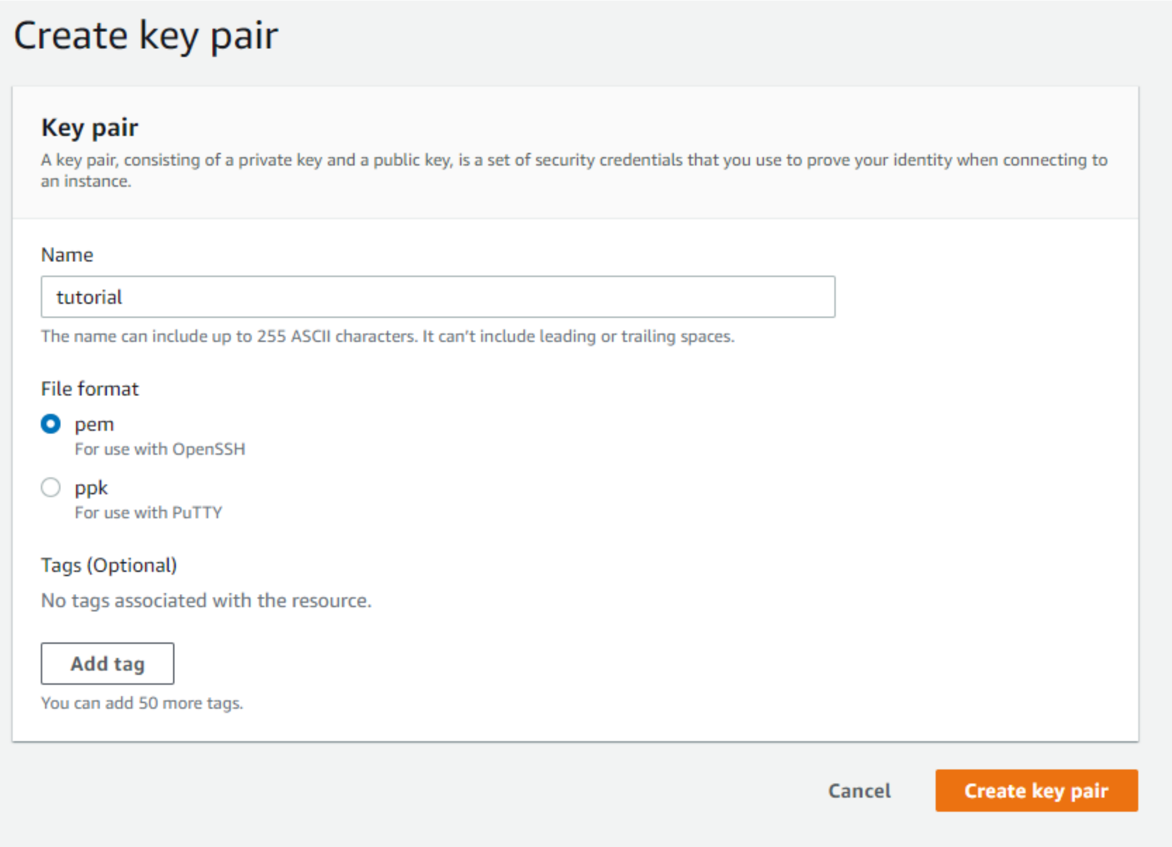


Key pairs (1)

Filter key pairs

□	Name	Fingerprint	ID
□	Test	fe:16:34:58:7d:80:5a:11:92:e2:f4:6e:2f...	key-0a68bc45ac4d25d8f

- Make sure **.pem** is selected



Create key pair

Key pair

A key pair, consisting of a private key and a public key, is a set of security credentials that you use to prove your identity when connecting to an instance.

Name

tutorial

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

File format

● pem
For use with OpenSSH

○ ppk
For use with PuTTY

Tags (Optional)

No tags associated with the resource.

You can add 50 more tags.

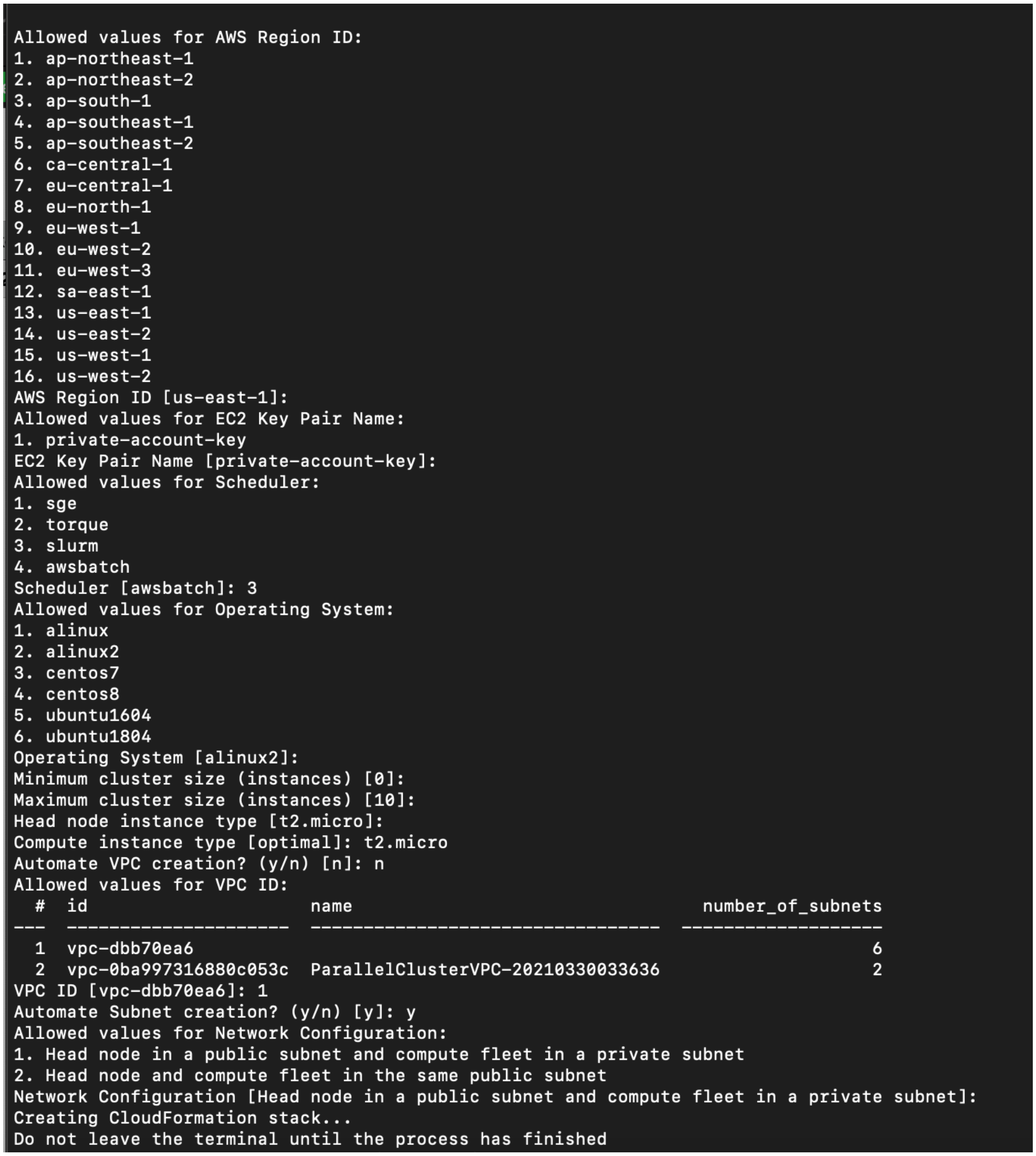
- Move the downloaded key file to your home directory(whatever directory the virtual environment is in)
- In the terminal change the permission of the key to read only access by running the command `chmod 400 <your-key>.pem`
- Now we can create and run jobs using the Slurm Scheduler or the AWS Batch Scheduler

Slurm - Configuration and Running Jobs:

AWS offers a setup with the Slurm scheduler, which is what is used by JMU's own cluster. Before starting this section, make sure you are in your virtual environment for AWS Parallel Cluster. Similarly to AWS Batch this will automatically scale based on job requirements so it is recommended running `sbatch` jobs so that you do not have to wait for each job to deploy individually.

Slurm Configuration:

- **NOTE**: If you have ran pcluster configure before, and you want to replicate the setup as the **exact same as last time you did,** you can just skip pcluster configure set up and go right to creating it
  - If you are not sure it does not hurt anything to go through the configuration process again
- In the terminal and run "`pcluster configure`" and enter the following (only type in the **bolded text for this section** normal font is just an explanation)
  - Region ID - **13**
  - EC2 Pair Name - **Number associated with .pem file created earlier**
  - Scheduler - **3**
  - Operating System - **2**
  - Minimum cluster size - **0**
  - Maximum cluster size - **10**
  - Head node instance type - **t2.micro**
  - Compute instance type - **t2.micro**
  - Automate VPC creation - **n**
  - Select VPC - **1**
  - Automate subnet creation - **y**
  - Values for network creation - **1**



```
Allowed values for AWS Region ID:
1. ap-northeast-1
2. ap-northeast-2
3. ap-south-1
4. ap-southeast-1
5. ap-southeast-2
6. ca-central-1
7. eu-central-1
8. eu-north-1
9. eu-west-1
10. eu-west-2
11. eu-west-3
12. sa-east-1
13. us-east-1
14. us-east-2
15. us-west-1
16. us-west-2
AWS Region ID [us-east-1]:
Allowed values for EC2 Key Pair Name:
1. private-account-key
EC2 Key Pair Name [private-account-key]:
Allowed values for Scheduler:
1. sge
2. torque
3. slurm
4. awsbatch
Scheduler [awsbatch]: 3
Allowed values for Operating System:
1. alinux
2. alinux2
3. centos7
4. centos8
5. ubuntu1604
6. ubuntu1804
Operating System [alinux2]:
Minimum cluster size (instances) [0]:
Maximum cluster size (instances) [10]:
Head node instance type [t2.micro]:
Compute instance type [optimal]: t2.micro
Automate VPC creation? (y/n) [n]: n
Allowed values for VPC ID:
  #  id                     name                               number_of_subnets
---  ---------------------  ---------------------------------  -------------------
  1  vpc-dbb70ea6                                                                6
  2  vpc-0ba997316880c053c  ParallelClusterVPC-20210330033636                    2
VPC ID [vpc-dbb70ea6]: 1
Automate Subnet creation? (y/n) [y]: y
Allowed values for Network Configuration:
1. Head node in a public subnet and compute fleet in a private subnet
2. Head node and compute fleet in the same public subnet
Network Configuration [Head node in a public subnet and compute fleet in a private subnet]:
Creating CloudFormation stack...
Do not leave the terminal until the process has finished
```

- After this you should be able to see the stack being created on the aws cloudformation console - this may take a minute or two

Creating the cluster:

- After this run “`pcluster create <your-cluster-name>`”
    - EX: `pcluster create tutorial`
- Log on to the login node with “`pcluster ssh <cluster name> -i /path/to/keyfile.pem`”

- On your first login your computer may ask you to confirm the ECDSA fingerprint of the head node.

Running Jobs with Slurm:

Because the Slurm scheduler is the same as the JMU Cluster the cluster reference guide will be helpful and can be found here: https://w3.cs.jmu.edu/lam2mo/cs470/cluster.html . Running commands using `sbatch` will be the most efficient use of the cluster because of the overhead for the autoscaling. Doing this will allow jobs to run in background and progress can be checked using `squeue`.

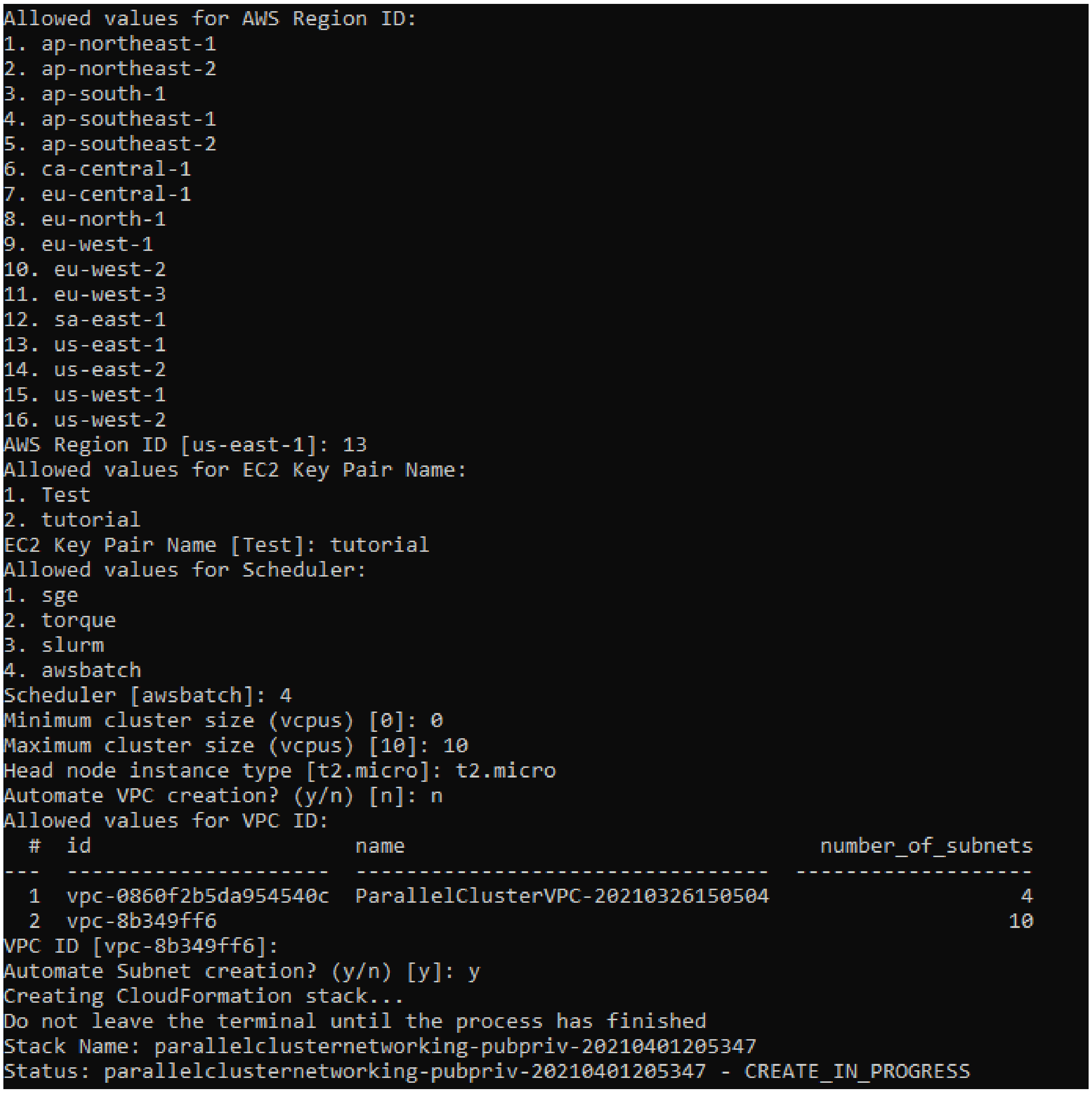
AWS Batch - Configuration and Running Jobs:

AWS Batch is Amazon’s own dynamically scaling job scheduler. Before starting this section make sure you are in your AWS Parallel Cluster virtual environment. More information about AWS Batch can be found here: https://aws.amazon.com/batch/.

IMPORTANT: Although the setup for AWS Batch is included for instructional purposes running jobs with this configuration **will** cause a charge to your AWS Account, for running jobs we recommend using the slurm configuration as it is compliant with the free tier of AWS.

Parallel Cluster Configuration:

- **NOTE**: If you have ran pcluster configure before, and you want to replicate the setup as the **exact same as last time you did** you can just skip pcluster configure set up and go right to creating it
  - If you are not sure it does not hurt anything to go through the configuration process again
- In the terminal and run “`pcluster configure`” and enter the following (only type in the **bolded text for this section** normal font is just an explanation)
  - Region ID - **13**
  - EC2 Pair Name - **Number associated with .pem file created earlier**
  - Scheduler - **4**
  - Minimum cluster size - **0**
  - Maximum cluster size - **10**
  - Head node instance type - **t2.micro**
  - Automate VPC creation - **n**
  - Select VPC - **1**
  - Automate subnet creation - **y**



```
Allowed values for AWS Region ID:
1. ap-northeast-1
2. ap-northeast-2
3. ap-south-1
4. ap-southeast-1
5. ap-southeast-2
6. ca-central-1
7. eu-central-1
8. eu-north-1
9. eu-west-1
10. eu-west-2
11. eu-west-3
12. sa-east-1
13. us-east-1
14. us-east-2
15. us-west-1
16. us-west-2
AWS Region ID [us-east-1]: 13
Allowed values for EC2 Key Pair Name:
1. Test
2. tutorial
EC2 Key Pair Name [Test]: tutorial
Allowed values for Scheduler:
1. sge
2. torque
3. slurm
4. awsbatch
Scheduler [awsbatch]: 4
Minimum cluster size (vcpus) [0]: 0
Maximum cluster size (vcpus) [10]: 10
Head node instance type [t2.micro]: t2.micro
Automate VPC creation? (y/n) [n]: n
Allowed values for VPC ID:
  #  id                     name                                number_of_subnets
---  ---------------------  --------------------------------  -------------------
  1  vpc-0860f2b5da954540c  ParallelClusterVPC-20210326150504                    4
  2  vpc-8b349ff6                                                               10
VPC ID [vpc-8b349ff6]:
Automate Subnet creation? (y/n) [y]: y
Creating CloudFormation stack...
Do not leave the terminal until the process has finished
Stack Name: parallelclusternetworking-pubpriv-20210401205347
Status: parallelclusternetworking-pubpriv-20210401205347 - CREATE_IN_PROGRESS
```

- After this you should be able to see the stack being created on the aws cloudformation console - this may take a few minutes

Creating the cluster:

- After this run "`pcluster create <your-cluster-name>`"
  - EX: `pcluster create tutorial`
- Log on to the login node with "`pcluster ssh <cluster name> -i /path/to/keyfile.pem`"
  - On your first login your computer may ask you to confirm the ECDSA fingerprint of the head node.
- Run `awsbhosts` and check that the instance type is t2.micro

Running a sample script:

- **IMPORTANT!** Running jobs on AWS Batch **will** cause a m4.large instance to run, which **is not** in the free tier of AWS
- Create a dummy script or copy this one:

```
#!/bin/bash

sleep 30
echo "Hello $1 from $HOSTNAME"
echo "Hello $1 from $HOSTNAME" >
"/shared/secret_message_for_${1}_by_${AWS_BATCH_JOB_ID}"

awsbsub -jn hello -cf hello.sh tutorial
```

- Run awsbstat to see the status of the job:
    - `awsbstat <job-id>`
    - Ex) `awsbstat 7e7e1777e177e10e-efe19239 131-fefss`
- Run `awsbout <job-id>` to see the output of the completed job

Troubleshooting and Notes:

- Ensure you are using the most recent version of AWS CLI
- If you are having trouble with python3 (EX: Some version of python 2 is being used instead)
    - For windows uninstall the old version and remove it from your path
    - For OS/Linux it should be pretty straightforward using homebrew/apt-get respectively
        - There are lots of resources online for how to do both of these
- If you have a previous pcluster configuration and you want to create a new file you will have to delete the previous configuration in order to write a new one
- When running pcluster configure if you get the message there is no credentials make sure that:
    - You have an ec2 keypair(.pem) you generated
    - This file is in the directory you activated your virtual environment in
- If there is an error message for access being denied make sure you did the chmod 400 command on your .pem file
- Root users have limited capability on the compute nodes so make sure that you are using the access key and secret key ID from the admin user that was created when doing AWS configure

- If at any point you aren't sure if you had entered the wrong AWS credentials during AWS configure or the wrong cluster configuration during pcluster configure you can run the command again to reset what you did.
- If you get an OPENSSL_LOCAL_CERTIFICATE issue most likely there was an issue with your credentials, regenerate them and mark the old ones inactive.
- To delete any parallel cluster you have created in your aws console search and click on the link for CloudFormation - you should see a list of your stacks
    - Look for the stack that has a name "parallelcluster-<your cluster name>", select it and press delete
    - This may take a few minutes because it must delete all of its nested substacks first.