Deploying an Elastic HPC Cluster

Build an elastic HPC cluster in minutes using AWS ParallelCluster

October 2019

Notices

Contents

Abstract

AWS ParallelCluster is an open source cluster management tool supported by AWS that is built on the open source CfnCluster project. It helps you to deploy and manage High Performance Computing (HPC) clusters in the AWS Cloud. This tutorial shows you how to set up, install, configure, and deploy AWS ParallelCluster. It also includes instructions to power down the cluster when it is not needed.

Introduction

Imagine a high performance computing (HPC) cluster that delivers the capabilities of a supercomputer and can be deployed in less than 15 minutes. AWS ParallelCluster is an open source tool published by AWS that provides a fully-elastic HPC cluster in the cloud. AWS ParallelCluster constructs an HPC environment resembling conventional HPC clusters but with the added benefit of being scalable. Jobs are submitted to a queue. Nodes spin up as needed. Jobs are automatically launched. As nodes become idle, they are automatically shut down. Once created, the cloud-based master node provides access to standard HPC tools such as schedulers, shared storage, and a Message Passing Interface (MPI) environment. The master node maintains the scheduler and the running environment while the compute nodes spin up and down as jobs are submitted to the queue.

This tutorial provides the steps necessary to install and deploy AWS ParallelCluster, configure the environment, run a simple parallel *mpi_hello_world* job, and shut down the cluster. This tutorial can be completed in less than one hour and, if the defaults are used when running the tutorial, is expected to cost less than one dollar.

AWS ParallelCluster is most often installed on a local computer running macOS or Linux. From the local computer, AWS ParallelCluster launches an HPC cluster that includes a master instance and on-demand compute nodes running on Amazon Elastic Compute Cloud (Amazon EC2). If a local Mac or Linux computer is not available, then AWS ParallelCluster can be run from an EC2 instance running Linux. For more information, see AWS ParallelCluster.

Step 1: Set Up Prerequisites for AWS ParallelCluster

Before you begin this exercise you must have an AWS account, an Amazon EC2 key pair, and have the AWS Command Line Interface (CLI) installed and configured. You must also have Python and pip installed on the launch computer. If you meet these prerequisites, you can proceed to *Step 2: Install AWS ParallelCluster*.

Sign in to AWS

If you already have an AWS account, sign in to your account and proceed to the *Create and Amazon EC2 Key Pair* section. Otherwise, follow these steps to create a new AWS account:

1. Access https://aws.amazon.com/.
2. Choose **Create an AWS Account**.
3. Follow the instructions.

Create an Amazon EC2 Key Pair

You must have an Amazon EC2 key pair to connect to the nodes in your cluster over a secure channel using the Secure Shell (SSH) protocol. If you already have a key pair that you want to use, you can skip this step. If you don't have a key pair, follow these steps to create a key pair.

Set Up the AWS CLI

Installation procedures vary depending on your operating system and environment. Options include an MSI installer, a bundled installer, and pip. Complete the following steps to install and configure the AWS CLI:

1. Installing the AWS CLI.
2. Configuring the AWS CLI.
3. Using the AWS CLI.

NOTE: The AWS CLI makes API calls to services over HTTPS. Outbound connections on TCP port 443 must be enabled in order to perform calls.

Install Python & Pip on the Launch Computer

AWS ParallelCluster is written in Python and is installed with pip (the Python installation package). Pip is installed by default when you use the following versions of Python:

- Python version 2 that is equal to or above v2.7.9
- Python version 3 that is equal to or above v3.4.

If Python is not already installed on the launch computer, go to the Python Software Foundation site for instructions and installation package.

To verify whether you have Python installed on your computer (and the version of Python), enter the following at the command prompt:

```
$ python --version
```

To verify whether you have pip installed on your computer (and the version of pip), enter the following at the command prompt:

```
$ pip --version
```

Step 2: Install AWS ParallelCluster

To install AWS ParallelCluster in the launch computer, enter the following in the command line prompt:

```
$ sudo pip install --upgrade aws-parallelcluster
```

Step 3: Configure AWS Credentials

To set up the correct permissions, you install the AWS CLI.

NOTE: You do not have to install the AWS CLI if you're running on an Amazon EC2 instance with an associated AWS Identity and Access Management (IAM) role.

1. To install the AWS CLI, run this command on your computer.

```
$ sudo pip install awscli
```

2. To set up your IAM credentials, at the command prompt enter `aws configure`. For more information, see Configuring the AWS CLI in the *AWS Command Line User Guide*.

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]:
wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]:
```

3. To test whether the IAM credentials were configured correctly, enter the following command:

```
$ aws s3 ls
```

This command should return a list of Amazon S3 buckets in your account. If the command returns anything and doesn't give an error, your IAM credentials are configured correctly.

Step 4: Configure and Launch AWS ParallelCluster

You can use the following steps to create, configure, and launch your AWS ParallelCluster.

Create the Base AWS ParallelCluster Configure File

AWS ParallelCluster customization is specified with the *ParallelCluster* configuration file (config file). The config file is a simple text file with keyword entries. It is created with a pcluster configuration tool that creates a baseline config file for the user.

1. At a command prompt, enter the following:

```
$ pcluster configure
```

2. Accept the cluster template default option for the first entry by pressing **Enter**.
3. Enter an AWS Region identifier (ID) from the following list.
   Make sure to specify the acceptable value for your selected Region.

NOTE: This prompt cannot be left blank.

AWS Region ID acceptable values include:

- ap-south-1
- eu-west-3
- eu-west-2
- eu-west-1
- ap-northeast-2
- ap-northeast-1
- sa-east-1
- ca-central-1
- ap-southeast-1
- ap-southeast-2
- eu-central-1
- us-east-1
- us-east-2
- us-west-1
- us-west-2

4. At the `VPC Name [public]` prompt, press **Enter** to accept the default option.
5. At the `SSH Key Name` prompt, copy and paste the name of the Amazon EC2 keypair created earlier. Do **NOT** include the `.pem` suffix.
6. From the list of VPC IDs, copy and paste a `VPC ID`.

7. From the list of Subnet IDs, copy and paste a `VPC Subnet ID`.
8. (Optional) To review the new config file and verify that the information is correct, enter the following command:

```
$ cat ~/.parallelcluster/config
```

The following is an example config file:

```
[aws]
aws_region_name = us-west-2

[cluster default]
vpc_settings = public
key_name = mysshkey

[vpc public]
master_subnet_id = subnet-0b14ad52
vpc_id = vpc-442b5d21

[global]
update_check = true
sanity_check = true
cluster_template = default

[aliases]
ssh = ssh {CFN_USER}@{MASTER_IP} {ARGS}
```

NOTE: Information on configuration settings are available in the AWS ParallelCluster documentation.

Customize the AWS ParallelCluster Config File

To customize the AWS ParallelCluster config file:

1. Open the config file in a text editor.
2. Scroll through the file and find the cluster default section.
3. Find the `key_name= mysshkey` entry. Add the following entries after it:

```
initial_queue_size = 3
max_queue_size = 3
maintain_initial_size = 3
```

> **NOTE:** These values result in a cluster with three compute nodes on launch and a maximum of three nodes in the cluster.

You can configure a wide range of capabilities by modifying the AWS ParallelCluster config file. For example, the `placement_group` keyword is used to create an Amazon EC2 placement group, which can result in increased performance for applications that require the lowest latency possible. For the example, in this tutorial, a placement group is not required. For more information, see the AWS ParallelCluster User Guide.

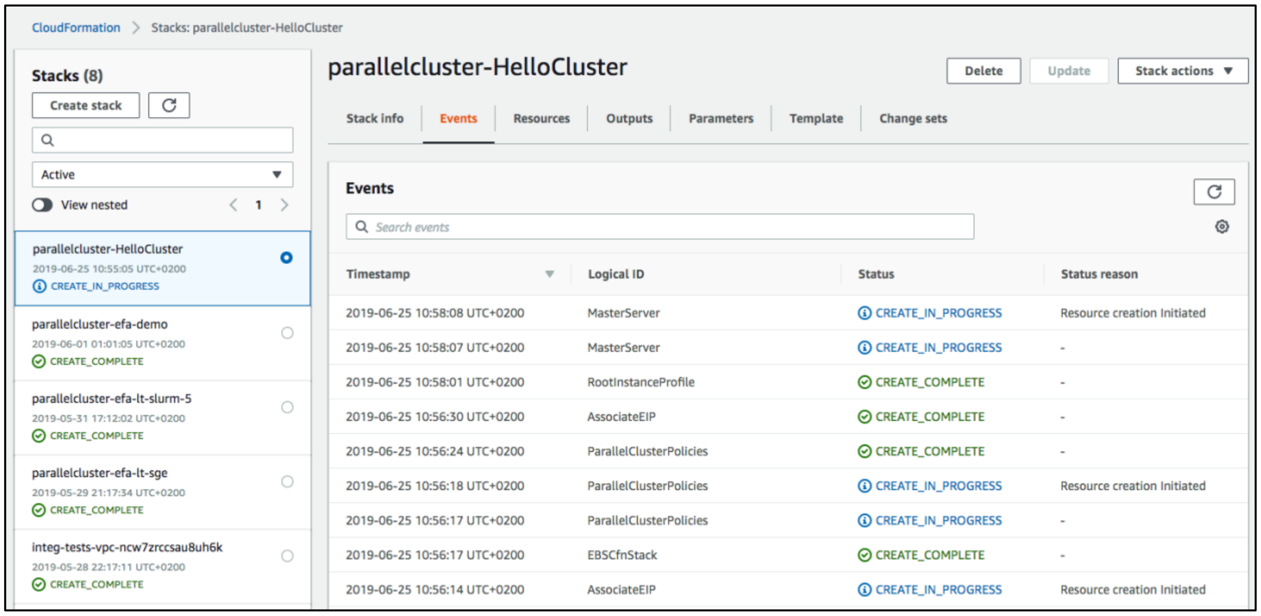
Launch the AWS ParallelCluster

To launch your cluster, at a command prompt, enter the following:

```
$ pcluster create HelloCluster
```

> **NOTE:** In this example, the cluster name is *HelloCluster*.

Full deployment of *HelloCluster* takes about 20 minutes, depending on the instance types you choose and the size of the Amazon Elastic Block Store (Amazon EBS) volumes you select. You can monitor the deployment progress from the AWS CloudFormation console events tab, as shown in *Figure 1*.


CloudFormation > Stacks: parallelcluster-HelloCluster

Stacks (8)

Stack	Created	Status
parallelcluster-HelloCluster	2019-06-25 10:55:05 UTC+0200	CREATE_IN_PROGRESS
parallelcluster-efa-demo	2019-06-01 01:01:05 UTC+0200	CREATE_COMPLETE
parallelcluster-efa-lt-slurm-5	2019-05-31 17:12:02 UTC+0200	CREATE_COMPLETE
parallelcluster-efa-lt-sge	2019-05-29 21:17:34 UTC+0200	CREATE_COMPLETE
integ-tests-vpc-ncw7zrccsau8uh6k	2019-05-28 22:17:11 UTC+0200	CREATE_COMPLETE

parallelcluster-HelloCluster

Events

Timestamp	Logical ID	Status	Status reason
2019-06-25 10:58:08 UTC+0200	MasterServer	CREATE_IN_PROGRESS	Resource creation Initiated
2019-06-25 10:58:07 UTC+0200	MasterServer	CREATE_IN_PROGRESS	-
2019-06-25 10:58:01 UTC+0200	RootInstanceProfile	CREATE_COMPLETE	-
2019-06-25 10:56:30 UTC+0200	AssociateEIP	CREATE_COMPLETE	-
2019-06-25 10:56:24 UTC+0200	ParallelClusterPolicies	CREATE_COMPLETE	-
2019-06-25 10:56:18 UTC+0200	ParallelClusterPolicies	CREATE_IN_PROGRESS	Resource creation Initiated
2019-06-25 10:56:17 UTC+0200	ParallelClusterPolicies	CREATE_IN_PROGRESS	-
2019-06-25 10:56:17 UTC+0200	EBSCfnStack	CREATE_COMPLETE	-
2019-06-25 10:56:14 UTC+0200	AssociateEIP	CREATE_IN_PROGRESS	Resource creation Initiated

Figure 1 – Example of the AWS CloudFormation console

When the cluster has successfully launched, a message similar to the following appears:

```
Status: CREATE_COMPLETE
MasterServer: RUNNING
MasterPublicIP: 3.212.159.171
ClusterUser: ec2-user
MasterPrivateIP: 172.31.29.209
```

NOTE: Both the public and private IP addresses for your cluster are provided. To disable the public IP address, set the `use_public_ips` flag to *False*. For more information, see the aws-parallelcluster GitHub repo.

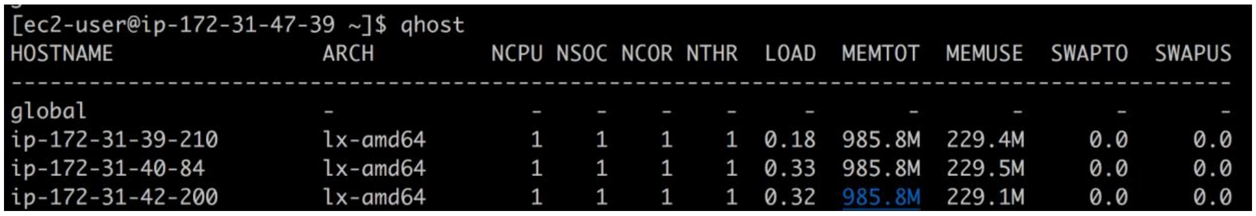
Sign In to Your Cluster

1. Sign in to your cluster with the public IP address and the SSH key.
   For example: `$ pcluster ssh HelloCluster –i ~/.ssh/mysshkey.pem`

2. To verify the current state of your cluster, enter the following command:

```
$ qhost
```

You'll see three hosts running. For example:



```
[ec2-user@ip-172-31-47-39 ~]$ qhost
HOSTNAME                ARCH         NCPU NSOC NCOR NTHR  LOAD  MEMTOT  MEMUSE  SWAPTO  SWAPUS
----------------------------------------------------------------------------------------------
global                  -               -    -    -    -     -       -       -       -       -
ip-172-31-39-210        lx-amd64        1    1    1    1  0.18  985.8M  229.4M     0.0     0.0
ip-172-31-40-84         lx-amd64        1    1    1    1  0.33  985.8M  229.5M     0.0     0.0
ip-172-31-42-200        lx-amd64        1    1    1    1  0.32  985.8M  229.1M     0.0     0.0
```

Step 5: Submit and Run a Simple Parallel MPI Job

In this exercise you submit and run a simple parallel MPI job. This process includes creating an executable, creating the job submittal file, and launching the job.

Create the *mpi_hello_world* Executable

1. Change to the following directory:

```
$ cd /shared
```

NOTE: A shared Network File System (NFS) mount is created for you and is available at /shared. The share is an EBS volume mounted and shared using NFS.

2. Create the following directory:

```
$ mkdir hw_work
```

3. Change to the directory you created:

```
$ cd hw_work
```

4. Copy and paste the following text into the file named *mpi_hello_world.c*

```
/*A Parallel Hello World Program*/
#include <stdio.h>
#include <mpi.h>
main(int argc, char **argv)
{
int a, node;
MPI_Init(&argc,&argv);
MPI_Comm_rank(MPI_COMM_WORLD, &node);
for ( a = 1; a < 5; a=a+1){
printf("Hello World from Node %d\n",node);
}
MPI_Finalize();
}
```

5. To compile the code, run the following command:

```
$ /opt/amazon/openmpi/bin/mpicxx mpi_hello_world.c -o hw.x
```

You have created the *mpi_hello_world* executable.

Create the Job Submittal File

To create the job submittal file, copy and paste the following text into the *hw.job* file.

```
#!/bin/sh
#$ -cwd
#$ -N helloworld
#$ -pe mpi 3
#$ -j y
date
/opt/amazon/openmpi/bin/mpirun ./hw.x > hello_all.out
```

Launch the Job

1. To submit the job, at the command prompt, enter the following:

```
$ qsub hw.job
Your job 1 ("helloworld") has been submitted
```

NOTES:

Before the job completes, you can check the status by entering `qstat` at the command prompt.

When the job completes, two output files are created: *hello_all.out* and *helloworld.o1*

2. To see the contents of the *hello_all.out* file, run the following command:

```
$ cat hello_all.out
```

When the command completes, you should see the following output:

```
$ cat hello_all.out
```

```
Hello World from Node 0
Hello World from Node 0
Hello World from Node 0
Hello World from Node 0
Hello World from Node 1
Hello World from Node 1
Hello World from Node 1
Hello World from Node 1
Hello World from Node 2
Hello World from Node 2
Hello World from Node 2
Hello World from Node 2
```

3. To see the contents of the *hello_world.o1* file, run the following command:

```
$ cat helloworld.o1
```

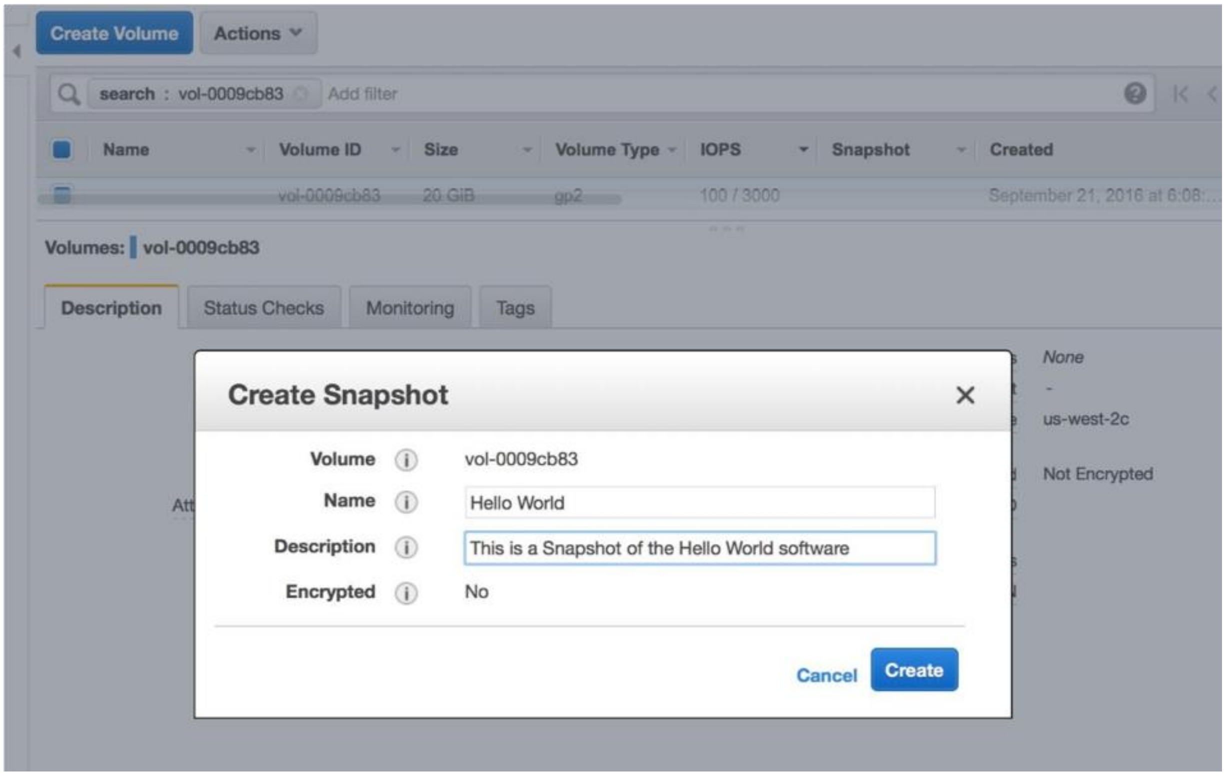
   When the command completes, you should see the following output:

```
$ cat helloworld.o1
Fri Sep 2 04:43:00 UTC 2016
```

4. To log out of the master instance, enter **exit** or press **Ctrl-d**.

Step 6: Create an Amazon EBS Volume Snapshot for Cluster Reusability

When you set up clusters, it is common to install large, frequently-used HPC applications to the shared drive—*/shared*—that resides on an Amazon EBS volume. By creating a snapshot of this EBS volume (see Figure 2), you can deploy the same pre-configured software on future clusters. The following is an example of this process.



search : vol-0009cb83 Add filter

Name	Volume ID	Size	Volume Type	IOPS	Snapshot	Created
	vol-0009cb83	20 GiB	gp2	100 / 3000		September 21, 2016 at 6:08:...

Volumes: vol-0009cb83

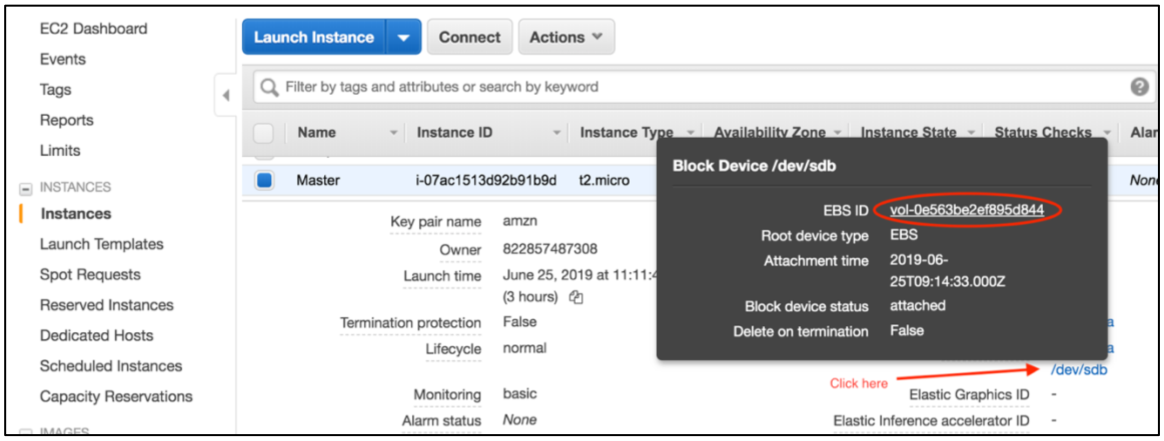
Create Snapshot

Volume	vol-0009cb83
Name	Hello World
Description	This is a Snapshot of the Hello World software
Encrypted	No

Figure 2 – Example of creating an EBS snapshot in the console

1. Go to the Amazon EC2 console and select the master instance with the name *Master*.
2. Scroll down to the block devices section of the instance description and click **/dev/sdb**.
   The Block Device dialog box appears with the details for this device.

3. To see the volume dashboard, click the **volume ID**.
   For example, vol-xxxxxxxx.



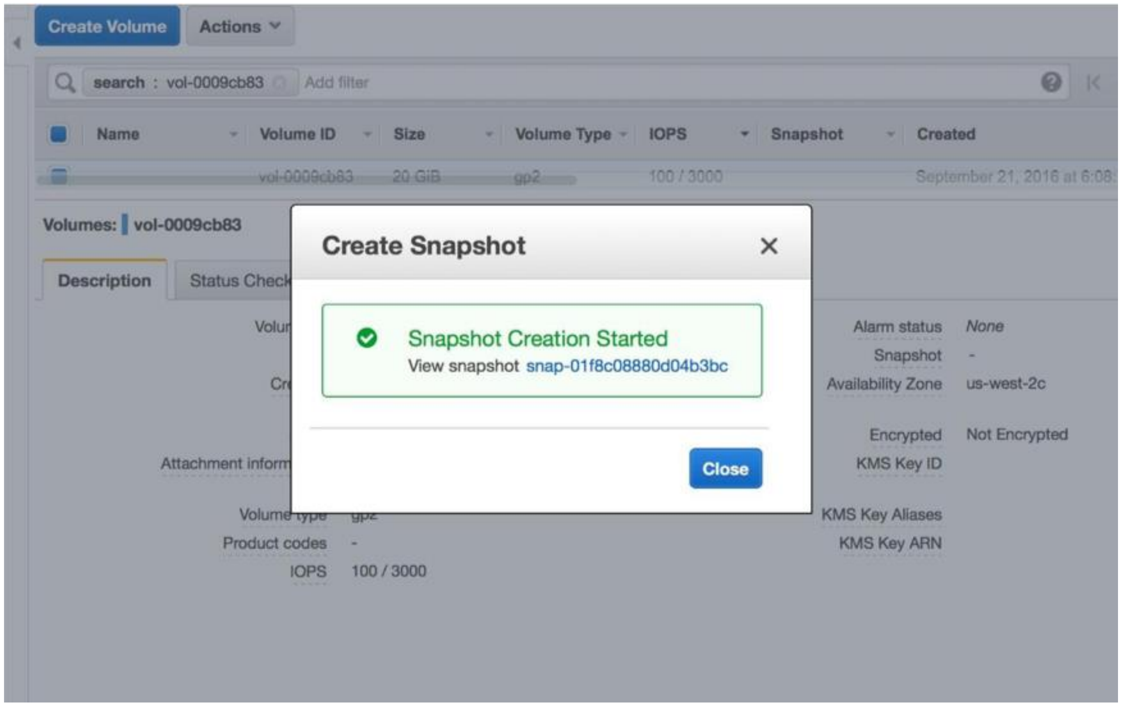
Name	Instance ID	Instance Type
Master	i-07ac1513d92b91b9d	t2.micro

Key pair name	amzn
Owner	822857487308
Launch time	June 25, 2019 at 11:11:4 (3 hours)
Termination protection	False
Lifecycle	normal
Monitoring	basic
Alarm status	None
Elastic Graphics ID	-
Elastic Inference accelerator ID	-

Block Device /dev/sdb

EBS ID	vol-0e563be2ef895d844
Root device type	EBS
Attachment time	2019-06-25T09:14:33.000Z
Block device status	attached
Delete on termination	False

4. From the **Action** drop-down list, select **Create Snapshot**.
5. Create a tag with the **Name** key and a value of **Hello World,** and enter the following description: **This is a Snapshot of the Hello World software**.
6. Click **Create Snapshot**.
   The Create Snapshot dialog box appears, with the snapshot ID number.
   In this example, the ID number is: *snap-0896bea72d42813f3*
7. Save the snapshot ID number to use in Step 8. You can copy it and save it in a text file.



Name	Volume ID	Size	Volume Type	IOPS	Snapshot	Created
	vol-0009cb83	20 GiB	gp2	100 / 3000		September 21, 2016 at 6:08

Create Snapshot

Snapshot Creation Started
View snapshot snap-01f8c08880d04b3bc

Volume type	gp2
Product codes	-
IOPS	100 / 3000
Alarm status	None
Snapshot	-
Availability Zone	us-west-2c
Encrypted	Not Encrypted
KMS Key ID	
KMS Key Aliases	
KMS Key ARN	

8. Edit the ParallelCluster config file (*~/.parallelcluster/config*) and add the following entries:

```
maintain_initial_size = 3
ebs_settings = helloebs
```

9. Add the following entries to the end of the config file:

```
[ebs helloebs]
ebs_snapshot_id = snap-XXXXXXXXXXXXXXXX (using your EBS snapshot
ID)
```

Note: You can launch this config file as a new cluster and the previously created volume and software is automatically available on the shared drive: */shared*.

Step 7: Delete and Clean Up the Cluster

When you are finished with the AWS ParallelCluster, you can clean up the related resources and delete the cluster.

1. To see a list of running clusters, open a command prompt on the launch computer and enter the following command:

```
$ pcluster list
HelloCluster   CREATE_COMPLETE    2.4.0
```

2. To clean up and remove the cluster, enter the following command:

```
$ pcluster delete HelloCluster
```

If the cluster was successfully deleted, the following message appears:

```
Deleting: HelloCluster
Status: DynamoDBTable - DELETE_COMPLETE
Stack with id  parallelcluster-HelloCluster does not exist
```

The cluster and all AWS cluster-related infrastructure is removed from AWS.

To restart the cluster when you need it again, run the following command:

```
$ pcluster create HelloCluster2
```

The cluster is created with *mpi_hello_world*, which you already installed on the shared filesystem, */shared*.

When you are finished with the snapshot you created in Step 5, you can delete it from the snapshot console.

1. At the left side of the Amazon EC2 dashboard, select the **EBS/Snapshot** tab.
2. Find the snapshot you created in Step 5 and delete it.

Conclusion

This tutorial provided a low cost method to help you install, configure, deploy, and remove AWS ParallelCluster. By following the steps in this tutorial, you learned how to configure AWS credentials, launch and access AWS ParallelCluster, and submit a simple job. You learned how to manage a fully-elastic HPC cluster in the cloud. Because you can use Amazon EBS volumes to preserve applications between different clusters, you do not have to keep the cluster running constantly, which helps you to minimize costs.

Contributors

Contributors to this document include:

- Linda Hedges, Principal HPC Application Engineer, Amazon Web Services
- Sean Smith, Software Development Engineer, Amazon Web Services
- Matt Snell, Solutions Architect, Amazon Web Services

Further Reading

- AWS ParallelCluster User Guide
- Amazon EC2
- Amazon EC2 Key Pairs
- AWS Command Line Interface User Guide
- Python Software Foundation
- Amazon Elastic Compute Cloud User Guide for Linux Instances: Placement Groups
- AWS CloudFormation
- aws-parallelcluster GitHub repo
- Amazon EBS

Document Revisions

Date	Description
October 2019	Updated AWS Region IDs, the path to mpicxx, and updated other references to code and the UI to match the latest product changes. Other minor edits.
June 2019	Updated references to the service name (changed from CfnCluster to AWS ParallelCluster) and corresponding
September 2016	First publication