



# Distributed ML on AWS Cloud: Computing with CPUs and GPUs

Xin Wang @ BDAL (08/24/2021)

# Computation on Cloud

- Cloud Computing:
  - Instead of buying, owning, and maintaining physical data centers and servers, you can access technology services (such as computing power, storage, and databases) from a cloud provider like Amazon Web Services (AWS), Microsoft Azure, Google Cloud, ...
- Benefits:
  - Agility
  - Elasticity
  - Cost savings
  - Deploy globally in minutes
  - ...





# ML Computation on AWS Cloud

- Single machine computation:
  - CPU computation
  - GPU computation
- Distributed (Multiple) machine computation:
  - Multi-CPU computation
  - Multi-GPUs computation on single virtual node (skipped)
  - Multi-GPUs computation on multiple virtual nodes
- Demo
  - Web based
  - Command line automation via Boto library (skipped)

# AWS Amazon Elastic Compute Cloud (EC2)

- Step 1: Launch instances on EC2: <https://us-west-2.console.aws.amazon.com/ec2/v2/home>

The screenshot displays the AWS Management Console for the EC2 service in the US West (Oregon) region. The interface includes a left-hand navigation menu with options like EC2 Dashboard, Instances, Images, and Network & Security. The main content area is divided into several sections:

- Resources:** A summary of EC2 resources in the US West (Oregon) Region. It includes a table with the following data:

Resource Type	Count
Instances (running)	0
Dedicated Hosts	0
Elastic IPs	0
Instances	0
Key pairs	2
Load balancers	0
Placement groups	0
Security groups	4
Snapshots	3
Volumes	0
- Launch instance:** A section with a red box highlighting the "Launch instance" button. Below the button, it states: "Note: Your instances will launch in the US West (Oregon) Region".
- Service health:** A section showing the status of the service. It indicates that the service is operating normally in the US West (Oregon) region.
- Scheduled events:** A section showing no scheduled events for the US West (Oregon) region.
- Account attributes:** A section showing supported platforms, VPC, and other account settings.
- Explore AWS:** A section with links to learn more about GPU Powered ML Inference with G4dn, Run Apache Spark on EMR for Less, and Amazon EBS Backup and Restore.

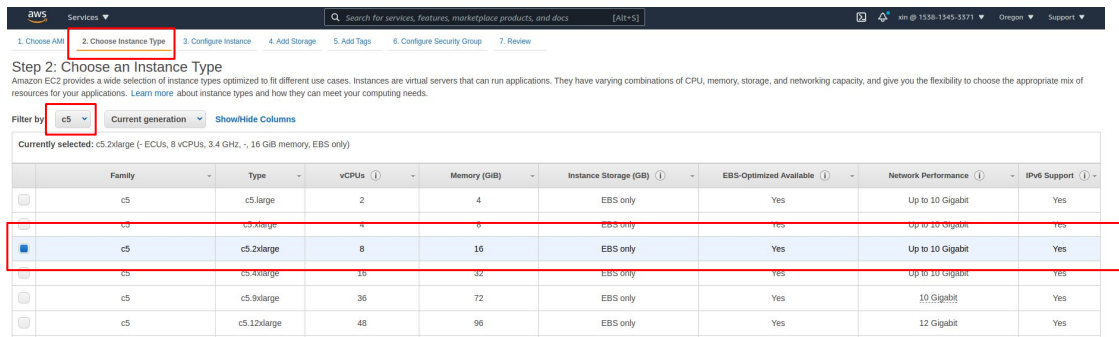
# AWS Amazon Elastic Compute Cloud (EC2)

- Step 2: Choose an Amazon Machine Image (AMI)
  - An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance.
  - For CPU applications, we use Ubuntu Server 16.04 LTS (HVM), SSD Volume Type; for GPU case, we use Deep Learning Base AMI (Ubuntu 16.04) Version 40.0.

The screenshot shows the AWS Management Console interface for the 'Choose an Amazon Machine Image (AMI)' step. The top navigation bar includes the AWS logo, 'Services', a search bar, and user information. The left sidebar shows the 'Quick Start (2)' menu with options for 'My AMIs (0)', 'AWS Marketplace (182)', and 'Community AMIs (893)'. The main content area displays a search bar with 'Ubuntu Server 16.04' and a list of two AMIs. The first AMI is 'Ubuntu Server 16.04 LTS (HVM) with SQL Server 2017 Standard' and the second is 'Ubuntu Server 16.04 LTS (HVM), SSD Volume Type'. The 'Select' button for the second AMI is highlighted with a red box. The right sidebar shows the 'Deep Learning Base AMI (Ubuntu 16.04) Version 40.0' with details about its architecture and drivers.

# AWS Amazon Elastic Compute Cloud (EC2)

- Step 3: Choose an Instance Type
  - Based on your purpose, AWS provides various instance types on <https://aws.amazon.com/ec2/instance-types/>
    - For CPU application, we use c5.2xlarge instance in our demo;
    - For GPU application, we use p3.2xlarge instance in our demo.

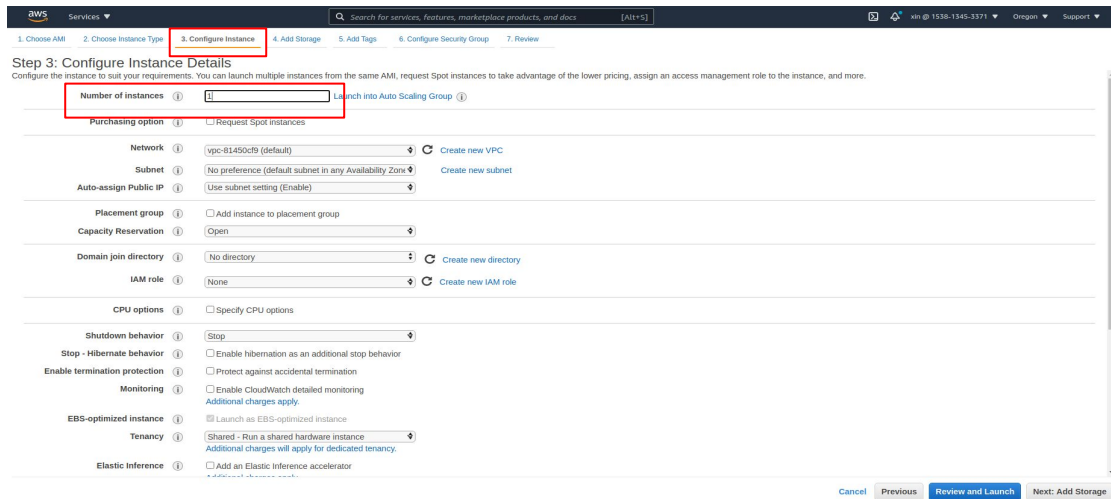


The screenshot shows the AWS Management Console interface for selecting an EC2 instance type. The 'Step 2: Choose an Instance Type' page is displayed, with the 'c5' filter selected. The 'c5.2xlarge' instance type is highlighted with a red box.

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GiB)	EBS-Optimized Available	Network Performance	IPv6 Support
<input type="checkbox"/>	c5	c5.large	2	4	EBS only	Yes	Up to 10 Gigabit	Yes
<input type="checkbox"/>	c5	c5.xlarge	4	8	EBS only	Yes	Up to 10 Gigabit	Yes
<input checked="" type="checkbox"/>	c5	c5.2xlarge	8	16	EBS only	Yes	Up to 10 Gigabit	Yes
<input type="checkbox"/>	c5	c5.4xlarge	16	32	EBS only	Yes	Up to 10 Gigabit	Yes
<input type="checkbox"/>	c5	c5.9xlarge	36	72	EBS only	Yes	10 Gigabit	Yes
<input type="checkbox"/>	c5	c5.12xlarge	48	96	EBS only	Yes	12 Gigabit	Yes

# AWS Amazon Elastic Compute Cloud (EC2)

- Step 4: Configure Number of instances
  - We use 1 instance for single machine computation, and 2 instances for distributed computation.



The screenshot displays the AWS Management Console interface for configuring an EC2 instance. The top navigation bar shows the AWS logo and a search bar. Below the navigation bar, the 'Step 3: Configure Instance Details' page is visible. The 'Number of instances' field is highlighted with a red box and set to 1. The 'Launch into Auto Scaling Group' checkbox is also checked. The page includes various configuration options such as Purchasing option, Network, Subnet, Auto-assign Public IP, Placement group, Capacity Reservation, Domain join directory, IAM role, CPU options, Shutdown behavior, Stop - Hibernate behavior, Enable termination protection, Monitoring, EBS-optimized instance, Tenancy, and Elastic Inference. The bottom of the page features buttons for 'Cancel', 'Previous', 'Review and Launch', and 'Next: Add Storage'.

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 of the lower pricing, assign an access management role to the instance, and more.

Number of instances  Launch into Auto Scaling Group ☒

Purchasing option ☐ Request Spot instances

Network  Create new VPC

Subnet  Create new subnet

Auto-assign Public IP

Placement group ☐ Add instance to placement group

Capacity Reservation

Domain join directory  Create new directory

IAM role  Create new IAM role

CPU options ☐ Specify CPU options

Shutdown behavior

Stop - Hibernate behavior ☐ Enable hibernation as an additional stop behavior

Enable termination protection ☐ Protect against accidental termination

Monitoring ☐ Enable CloudWatch detailed monitoring  
Additional charges apply.

EBS-optimized instance ☒ Launch as EBS-optimized instance

Tenancy  Additional charges will apply for dedicated tenancy.

Elastic Inference ☐ Add an Elastic Inference accelerator

Cancel Previous Review and Launch Next: Add Storage

# AWS Amazon Elastic Compute Cloud (EC2)

- Step 5: Configure Security Group

**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 can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☐ Create a new security group ☒ Select an existing security group

Security Group ID	Name	Description	Actions
sg-f0a248de	default	default VPC security group	<a href="#">Copy to new</a>
<b>sg-053abf4cac2d506f4</b>	<b>distributed_dl_starty</b>	<b>Distributed DL on AWS</b>	<a href="#">Copy to new</a>
sg-050eb4f97ec0e70de	ElasticMapReduce-master	Master group for Elastic MapReduce created on 2020-11-17T00:05:38.706Z	<a href="#">Copy to new</a>
sg-0db716d6290ced2d	ElasticMapReduce-slave	Slave group for Elastic MapReduce created on 2020-11-17T00:05:58.706Z	<a href="#">Copy to new</a>

Inbound rules for sg-053abf4cac2d506f4 (Selected security groups: sg-053abf4cac2d506f4)

Type	Protocol	Port Range	Source	Description
All TCP	TCP	0 - 65535	0.0.0.0/0	
All TCP	TCP	0 - 65535	:::0	
All traffic	All	All	0.0.0.0/0	
All traffic	All	All	:::0	
SSH	TCP	22	0.0.0.0/0	
SSH	TCP	22	76.106.18.7/32	

[Cancel](#) [Previous](#) [Review and Launch](#)



# AWS Amazon Elastic Compute Cloud (EC2)

- Step 6: Review, Create your SSH key pair, and Launch

The screenshot shows the AWS Management Console interface for launching an EC2 instance. The top navigation bar includes the AWS logo, 'Services', a search bar, and user information. The main content area displays the 'Step 7: Review Instance Launch' page. A red box highlights the '7. Review' step in the top navigation bar. Below the navigation bar, a warning message states: 'Your instance configuration is not eligible for the free usage tier. To launch an instance that's eligible for the free usage tier, check your AMI selection, instance type, configuration options, or storage devices. Learn more about free usage tier eligibility and usage restrictions.' Another warning message states: 'Improve your instances' security. Your security group, distribution, and other settings may be accessible from any IP address. We recommend that you can also open additional ports in your security group to facilitate access to your instance.' The main content area is divided into sections: 'AMI Details', 'Instance Type', and 'Security Groups'. The 'AMI Details' section shows 'Ubuntu Server 16.04 LTS (HVM), SSD Volume Type - ami-0dd273d'. The 'Instance Type' section shows 'c5.2xlarge' with 8 vCPUs and 16 GiB of memory. The 'Security Groups' section shows 'sg-053abf4cac2d506f4' with the name 'distributed\_el\_starly'. A modal dialog titled 'Select an existing key pair or create a new key pair' is open. The dialog contains a text area for 'Create a new key pair', a dropdown menu for 'Key pair type' (set to 'RSA'), a text input for 'Key pair name' (set to 'ec2\_key'), and a 'Download Key Pair' button. A note at the bottom of the dialog states: 'You have to download the private key file (\*.pem file) before you can continue. Store it in a secure and accessible location. You will not be able to download the file again after it's created.' The 'Launch Instances' button is highlighted in red.

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 instance and complete the launch process.

**Your instance configuration is not eligible for the free usage tier**  
To launch an instance that's eligible for the free usage tier, check your AMI selection, instance type, configuration options, or storage devices. Learn more about [free usage tier](#) eligibility and usage restrictions.

**Improve your instances' security.** Your security group, distribution, and other settings may be accessible from any IP address. We recommend that you can also open additional ports in your security group to facilitate access to your instance.

**AMI Details**

Ubuntu Server 16.04 LTS (HVM), SSD Volume Type - ami-0dd273d

Free tier eligible

Ubuntu Server 16.04 LTS (HVM), EBS General Purpose (SSD) Volume Type, Step 1

Root Device Type: ebs Virtualization type: hvm

**Instance Type**

Instance Type	ECUs	vCPUs	Memory (GiB)
c5.2xlarge	-	8	16

**Security Groups**

Security Group ID	Name
sg-053abf4cac2d506f4	distributed_el_starly

All selected security groups inbound rules

Type	Protocol	Port Range	Source	Description
------	----------	------------	--------	-------------

**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. Amazon EC2 supports ED25519 and RSA key pair types.

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](#).

Create a new key pair

**Key pair type**

☒ RSA ☐ ED25519

**Key pair name**

ec2\_key

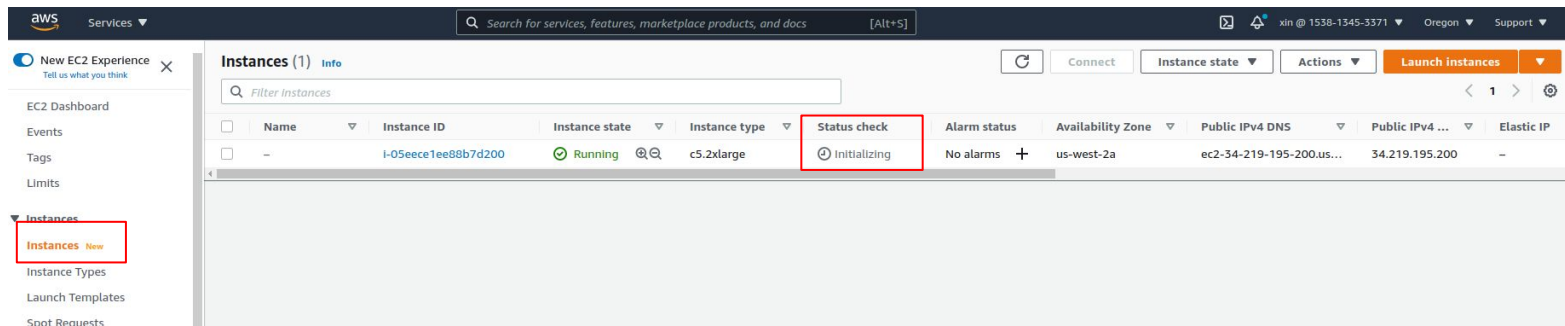
**Download Key Pair**

You have to download the **private key file** (\*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

**Launch Instances**

# AWS Amazon Elastic Compute Cloud (EC2)

- Step 7: View your Instance and wait for Initialing



The screenshot shows the AWS Management Console interface. On the left, the navigation menu includes 'New EC2 Experience', 'EC2 Dashboard', 'Events', 'Tags', 'Limits', and 'Instances'. The 'Instances' section is expanded, and 'Instances' is highlighted with a red box. The main content area displays a table of instances. The first instance, with ID 'i-05eece1ee88b7d200', is in the 'Running' state and its 'Status check' is 'Initializing', which is highlighted with a red box. The table also shows columns for Name, Instance ID, Instance state, Instance type, Alarm status, Availability Zone, Public IPv4 DNS, Public IPv4 address, and Elastic IP.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
-	i-05eece1ee88b7d200	Running	c5.2xlarge	Initializing	No alarms	us-west-2a	ec2-34-219-195-200.us...	34.219.195.200	-



<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
<input type="checkbox"/>	-	i-05eece1ee88b7d200	Running	c5.2xlarge	2/2 checks passed	No alarms	us-west-2a	ec2-34-219-195-200.us...	34.219.195.200	-

# Accessing Virtual Machines on your Command Line

- Step 8: SSH into your instance

Instances (1/1) Info

Filter Instances

Connect Instance state Actions Launch Instances

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
<input checked="" type="checkbox"/>	-	i-05eece1ee88b7d200	Running	c5.xlarge	2/2 checks passed	No alarms	us-west-2a	ec2-34-219-195-200.us...	\$4.219.195.200	-

**Connect to instance** Info

Connect to your instance i-01a34f61b2e2b521d using any of these options

EC2 Instance Connect Session Manager **SSH client** EC2 Serial Console

Instance ID  
i-01a34f61b2e2b521d

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is ec2\_key.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.  
`chmod 400 ec2_key.pem`
4. Connect to your Instance using its Public DNS:  
ec2-54-149-99-252.us-west-2.compute.amazonaws.com

Example:

```
ssh -i "ec2_key.pem" ubuntu@ec2-54-149-99-252.us-west-2.compute.amazonaws.com
```

**Note:** In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

```
ubuntu@ip-172-31-37-75: ~
starly@starly-Inspiron-3670:~$ ssh -i ~/Downloads/ec2_key.pem ubuntu@ec2-54-149-
99-252.us-west-2.compute.amazonaws.com
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 4.4.0-1128-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 of these updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-37-75:~$ ls
ubuntu@ip-172-31-37-75:~$
```



# Install Docker and images on Virtual Machines

- Step 9: Install Docker

- `curl -fsSL https://get.docker.com -o get-docker.sh`
- `sudo sh get-docker.sh`
- `sudo service docker start`
- `sudo usermod -a -G docker ubuntu`
- `sudo chmod 666 /var/run/docker.sock`

- Step 10: Download environment images (<https://hub.docker.com/u/starlyxxx>) or build images by Dockerfile

([https://drive.google.com/drive/folders/1YZ2\\_ThqX46Yvy7dEt9KsG52pvQfhXBq1?usp=sharing](https://drive.google.com/drive/folders/1YZ2_ThqX46Yvy7dEt9KsG52pvQfhXBq1?usp=sharing))

- CPU application environment image: `docker pull starlyxxx/dask-decision-tree-example`
- GPU application environment image: `docker pull starlyxxx/horovod-pytorch-cuda10.1-cudnn7`
- (Build from Dockerfile: `docker build -t <your-image-name> .`)



# Download source data on Virtual Machines

- Step 11: Download ML applications and data.
  - For privacy, we store the application code and data on AWS S3
  - Install aws cli and set aws credentials ([https://console.aws.amazon.com/iam/home?#security\\_credential](https://console.aws.amazon.com/iam/home?#security_credential)):

- `curl 'https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip' -o 'awscliv2.zip'`
- `unzip awscliv2.zip`
- `sudo ./aws/install`
- `aws configure set aws_access_key_id <your-access-key>`
- `aws configure set aws_secret_access_key <your-secret-key>`



# Download source data on Virtual Machines

- Step 11: Download ML applications and data.

- CPU application:

- `aws s3 cp s3://kddworkshop/ML based Cloud Retrieval Use Case.zip ./`
    - `(wget https://kddworkshop.s3.us-west-2.amazonaws.com/ML based Cloud Retrieval Use Case.zip)`
    - `unzip ML based Cloud Retrieval Use Case.zip`

- GPU application:

- `aws s3 cp s3://kddworkshop/MultiGpus-Domain-Adaptation-main.zip ./`
    - `(wget https://kddworkshop.s3.us-west-2.amazonaws.com/MultiGpus-Domain-Adaptation-main.zip)`
    - `aws s3 cp s3://kddworkshop/office31.tar.gz ./`
    - `(wget https://kddworkshop.s3.us-west-2.amazonaws.com/office31.tar.gz)`
    - `unzip MultiGpus-Domain-Adaptation-main.zip`
    - `tar -xzf office31.tar.gz`



# Running Docker Container for ML Computation

- Step 12: Run docker containers for CPU applications

- Single CPU:

- `docker run -it -v`

- `/home/ubuntu/ML based Cloud Retrieval Use Case:/root/ML based Cloud Retrieval Use Case starlyxxx/dask-decision-tree-example:latest /bin/bash`

- Multi-CPU:

- `docker run -it --network host -v`

- `/home/ubuntu/ML based Cloud Retrieval Use Case:/root/ML based Cloud Retrieval Use Case starlyxxx/dask-decision-tree-example:latest /bin/bash`

-v: shared filesystems

-it /bin/bash: interactive processes with bash

--network host: container shares the host's networking namespace

# Running Docker Container for ML Computation

- Step 13: Run docker containers for GPU applications

- Single GPU:

- `nvidia-docker run -it -v /home/ubuntu/MultiGpus-Domain-Adaptation-main:/root/MultiGpus-Domain-Adaptation-main -v /home/ubuntu/office31:/root/office31 starlyxxx/horovod-pytorch-cuda10.1-cudnn7:latest /bin/bash`

- Multi-GPUs:

- Add primary worker's public key to all secondary workers' `<~/ssh/authorized_keys>`
    - `sudo mkdir -p /mnt/share/ssh && sudo cp ~/.ssh/* /mnt/share/ssh`
    - Primary worker VM: `nvidia-docker run -it --network=host -v /mnt/share/ssh:/root/.ssh -v /home/ubuntu/MultiGpus-Domain-Adaptation-main:/root/MultiGpus-Domain-Adaptation-main -v /home/ubuntu/office31:/root/office31 starlyxxx/horovod-pytorch-cuda10.1-cudnn7:latest /bin/bash`
    - Secondary workers VM: `nvidia-docker run -it --network=host -v /mnt/share/ssh:/root/.ssh -v /home/ubuntu/MultiGpus-Domain-Adaptation-main:/root/MultiGpus-Domain-Adaptation-main -v /home/ubuntu/office31:/root/office31 starlyxxx/horovod-pytorch-cuda10.1-cudnn7:latest bash -c "/usr/sbin/sshd -p 12345; sleep infinity"`





# ML Computation on Docker Container

- Step 14: Run ML CPU application:
  - Single CPU:
    - ```
cd ML_based_Cloud_Retrieval_Use_Case/Code && /usr/bin/python3.6  
ml_based_cloud_retrieval_with_data_preprocessing.py
```
  - Multi-CPU:
    - Run dask cluster on both VMs in background:
      - VM 1: 

```
dask-scheduler & dask-worker <your-dask-scheduler-address> &
```
      - VM 2: 

```
dask-worker <your-dask-scheduler-address> &
```
    - One of VMs:
      - ```
cd ML_based_Cloud_Retrieval_Use_Case/Code && /usr/bin/python3.6  
dask ml_based_cloud_retrieval_with_data_preprocessing.py  
<your-dask-scheduler-address>
```



# ML Computation on Docker Container

- Step 15: Running ML GPU application:

- Single GPU:

- `cd MultiGpus-Domain-Adaptation-main`
    - `horovodrun --verbose -np 1 -H localhost:1 /usr/bin/python3.6 main.py --config DeepCoral/DeepCoral.yaml --data dir ../office31 --src domain webcam --tgt domain amazon`

- Multi-GPUs:

- Primary worker VM:

- `cd MultiGpus-Domain-Adaptation-main`
      - `horovodrun --verbose -np 2 -H <machine1-address>:1,<machine2-address>:1 -p 12345 /usr/bin/python3.6 main.py --config DeepCoral/DeepCoral.yaml --data dir ../office31 --src domain webcam --tgt domain amazon`

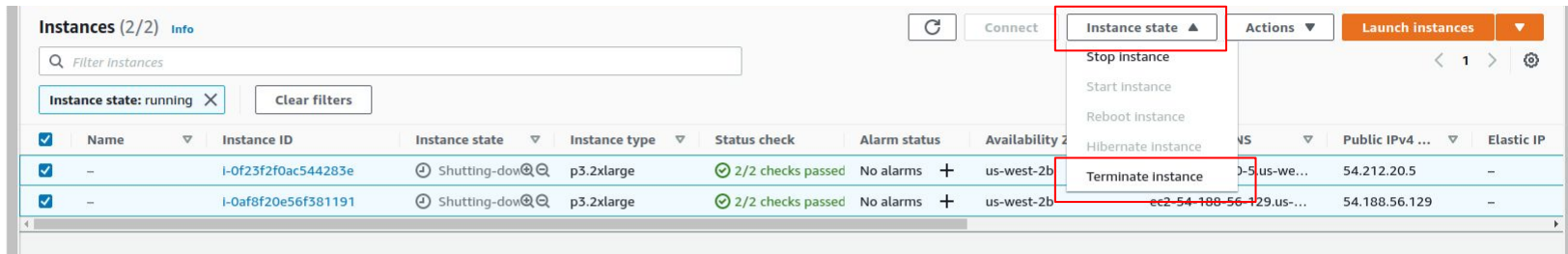
-np: number of processes

-H: list of hosts

-p: ssh port on all the hosts

# Terminate Virtual Machines

- Step 16: Terminate all VMs



The screenshot displays the AWS Management Console 'Instances' page. At the top, there's a header with 'Instances (2/2)' and an 'Info' link. Below this is a search bar labeled 'Filter instances' and a filter button set to 'Instance state: running'. A 'Clear filters' button is also present. On the right, there are buttons for 'Connect', 'Instance state' (which is open), 'Actions', and 'Launch instances'. The 'Instance state' dropdown menu is open, showing options: 'Stop Instance', 'Start Instance', 'Reboot Instance', 'Hibernate Instance', and 'Terminate Instance'. The 'Terminate Instance' option is highlighted with a red box. Below the menu is a table of instances. The table has columns: Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability zone, Public IPv4 address, and Elastic IP. Two instances are listed, both in the 'Shutting-down' state.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability zone	Public IPv4 address	Elastic IP
-	i-0f23f2f0ac544283e	Shutting-down	p3.2xlarge	2/2 checks passed	No alarms	us-west-2b	54.212.20.5	-
-	i-0af8f20e56f381191	Shutting-down	p3.2xlarge	2/2 checks passed	No alarms	us-west-2b	54.188.56.129	-



# Distributed ML on AWS Cloud: Demo