

Lab Steps

Task 1: Sign in to AWS Management Console

1. Click on the **Open Console** button, and you will get redirected to AWS Console in a new browser tab.
2. On the AWS sign-in page,

Leave the Account ID as default. Never edit/remove the 12-digit Account ID present in the AWS Console. Otherwise, you cannot proceed with the lab.

Now copy your **User Name** and **Password** in the Lab Console to the **IAM Username and Password** in AWS Console and click on the **Sign-in** button

3. Once Signed In to the AWS Management Console, make the default AWS Region as **US East (N. Virginia) us-east-1**.

Task 2: Create a Key Pair for the EC2 instances, inside the ECS Cluster

1. Make sure you are in the **N.Virginia** Region. Navigate to **EC2** Service by clicking on the **Services** menu in the top, then click on **EC2** Service in the **Compute** section.
2. In the left navigation pane (scroll down) within **Network & Security**, click on the **Key pairs**
3. To create a new key pair, click on the **Create key pair** button.
4. Fill in the details below:
 - a. Name: Enter **WhizKeyPair**
 - b. Key pair type : **RSA**
 - c. File format: **pem (Linux & Mac Users)** or **ppk (Windows users)**

- d. Leave other options as default.
- e. Click on the **Create key pair** button.

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

Key pair type [Info](#) RSA ED25519

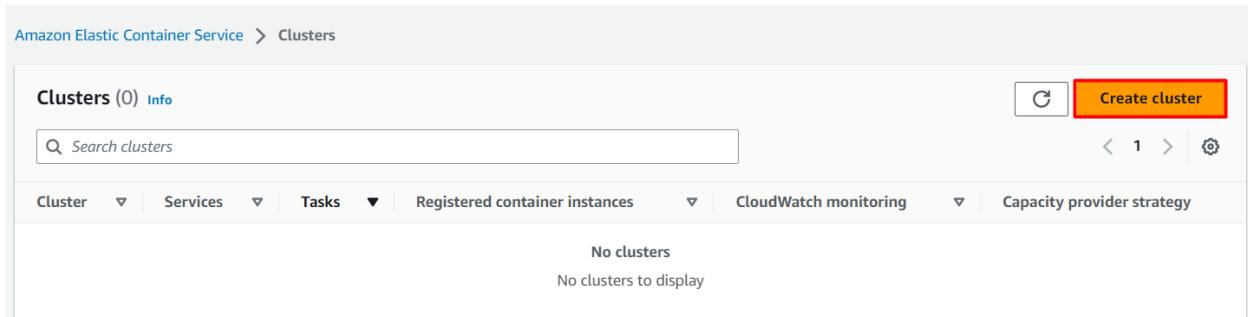
Private key file format
 .pem For use with OpenSSH
 .ppk For use with PuTTY

Tags - *optional*
No tags associated with the resource.
[Add new tag](#)
You can add up to 50 more tags.

[Cancel](#) **Create key pair**

Task 3: Launching an ECS Cluster

1. Make sure you are in the **N.Virginia** Region. Navigate to **Elastic Container Service** by clicking on the **Services** menu in the top, then click on **Elastic Container Service** in the **Containers** section.
2. On the left sidebar, click on the **Clusters** option present under the **Amazon ECS** section, then Click on the **Create cluster** button.



3. In Cluster configuration :

Cluster name : Enter **whiz**

Cluster configuration

Cluster name

whiz

There can be a maximum of 255 characters. The valid characters are letters (uppercase and lowercase), numbers, hyphens, and underscores.

4. In Infrastructure Section :

Uncheck AWS Fargate and **Check** Amazon EC2 Instances checkbox:

Auto Scaling group (ASG) : Select **Create new ASG**

Operating System/Architecture : Select **Amazon Linux 2**

EC2 instance type : Select **t2.micro**

Desired capacity :

Minimum : Enter **1**

Maximum : Enter **2**

SSH Key pair: Select **WhizKeyPair**

Auto Scaling group (ASG) | [Info](#)
Use Auto Scaling groups to scale the Amazon EC2 instances in the cluster.

Create new ASG

Provisioning model
Select a provisioning model for your instances

On-demand
With on-demand instances, you pay for compute capacity by the hour, with no long-term commitments or upfront payments.

Spot
Amazon EC2 Spot instances let you take advantage of unused EC2 capacity in the AWS cloud. Spot instances are available at up to a 90% discount compared to on-demand prices.

Operating system/Architecture
Choose the Windows operating system or Linux architecture for your instance.

Amazon Linux 2

EC2 instance type
Choose based on the workloads you plan to run on this cluster.

t2.micro
i386, x86_64
1 vCPU 1 GiB Memory

Desired capacity
Specify the number of instances to launch in your cluster.

Minimum **1** Maximum **2**

SSH Key pair
If you do not specify a key pair, you can't connect to the instances via SSH unless you choose an AMI that is configured to allow users another way to log in.

WhizKeyPair [Create a new key pair](#)

5. In Networking Section :

VPC: Select **Default VPC**

Subnets: Keep it **Default**

6. Security group :

Select **create new security group**

Security group name : Enter **MySG**

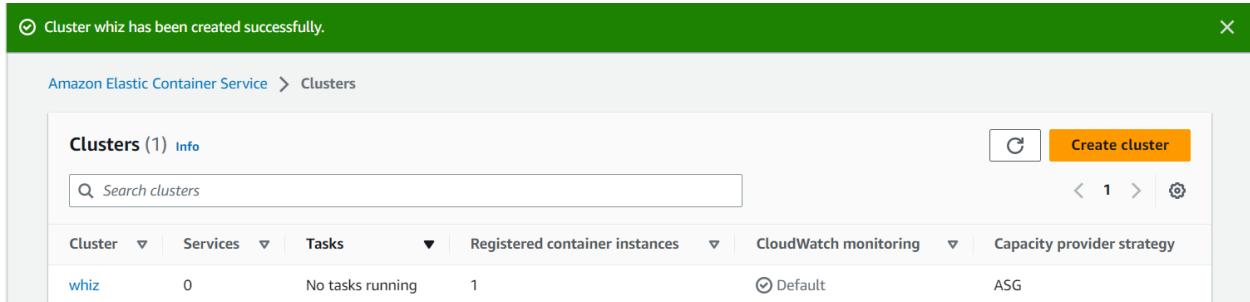
Security group description : Enter **MySG**

Inbound rules for security groups:

Type : Select **SSH**

Source : Select **Anywhere**

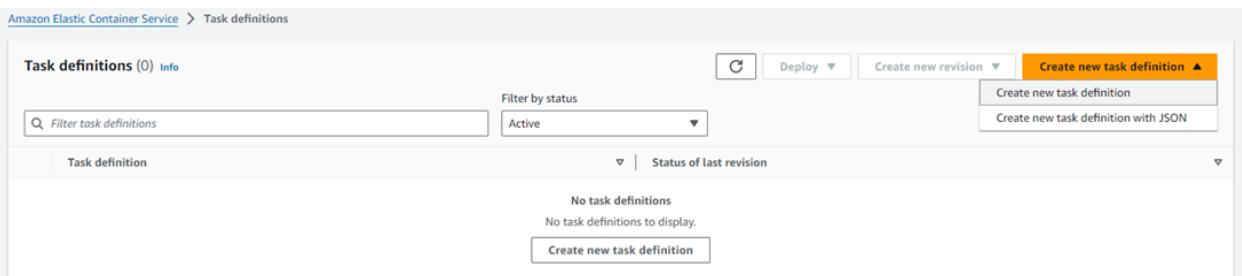
7. Keep rest things as default and click on **Create** button
8. **whiz** ECS Cluster will be created
9. It will take few minutes to provision the ECS Instance.



Task 4: Create Task Definitions

In this task, we are going to create a task definition for the cluster. A task definition provides a blueprint for creating tasks, which are the basic unit of work in ECS. Each task definition can define one or more containers that are run together on the same underlying EC2 instances or Fargate tasks.

1. On the left sidebar, click on the **Task Definitions** option present under the **Amazon ECS** section.
2. Click on the **Create new task definition** button.



3. In Task definition configuration:

Task definition family: Enter **ecs-demo**

Task definition configuration

Task definition family | [Info](#)
Specify a unique task definition family name.

ecs-demo

Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.

4. For Infrastructure Requirements :

Launch Type : **Remove AWS Fargate** and Select **Amazon EC2 Instances**

Task size :

CPU : Enter **.25 vCPU**

Memory : Enter **.5 GB**

▼ Infrastructure requirements

Specify the infrastructure requirements for the task definition.

Launch type | [Info](#)
Selection of the launch type will change task definition parameters.

AWS Fargate
Serverless compute for containers.

Amazon EC2 instances
Self-managed infrastructure using Amazon EC2 instances.

OS, Architecture, Network mode
Network mode is used for tasks and is dependent on the compute type selected.

Operating system/Architecture | [Info](#)

Linux/X86_64 ▾

Network mode | [Info](#)

awsvpc ▾

Task size | [Info](#)
Specify the amount of CPU and memory to reserve for your task.

CPU	Memory
.25 vCPU	.5 GB

5. For Container details :

Name: Enter **httpd**

Image URL: Enter **httpd:2.4**

6. For Port mappings :

Container port : Enter **80**

Protocol : Select **TCP**

App protocol : Select **HTTP**

Container - 1 [Info](#)

Essential container [Remove](#)

Container details
Specify a name, container image, and whether the container should be marked as essential. Each task definition must have at least one essential container.

Name	Image URI	Essential container
httpd	httpd:2.4	Yes ▾

Private registry [Info](#)
Store credentials in Secrets Manager, and then use the credentials to reference images in private registries.
 Private registry authentication

Port mappings [Info](#)
Add port mappings to allow the container to access ports on the host to send or receive traffic. Any changes to port mappings configuration impacts the associated service connect settings.

Container port	Protocol	Port name	App protocol
80	TCP ▾	httpd-80-tcp	HTTP ▾

[Add more port mappings](#) [Remove](#)

8. Keep rest things as default and click on the **Create** button

9. Task Definition **ecs-demo** is now created.

Task 5: Create a service and start HTTPD container in ECS

In this task, we are going to create a service and start the HTTPD container. In AWS ECS (Elastic Container Service), a service is a long-running task that ensures that a specified number of instances of a task definition are running and maintained in an ECS cluster. Services allow you to define the desired state of your tasks and automatically handle task placement, scaling, and recovery.

1. On the left sidebar, click on the **Task definitions** option present under the **Amazon ECS** section.
2. Select **ecs-demo** and click on **Deploy** and click **Create service** button.

The screenshot shows the 'Task definitions' page in the AWS ECS console. There is one task definition named 'ecs-demo' listed as 'ACTIVE'. The page has a search bar at the top, a filter for 'Active' status, and buttons for 'Deploy', 'Create new revision', and 'Create new task definition'.

3. Existing cluster : Select **whiz** cluster
4. Compute options : Select **Launch Type**

Launch type: Select **EC2**

The screenshot shows the 'Environment' configuration page. The 'Existing cluster' dropdown is set to 'whiz'. The 'Compute configuration (advanced)' section contains two options: 'Capacity provider strategy' and 'Launch type'. The 'Launch type' option is selected and highlighted with a blue box. Below it, a dropdown menu for 'Launch type' is open, showing 'EC2' as the selected option, which is also highlighted with a red box.

5. In Deployment configurations :

Service name: Enter **httpd**

Service type: Select **REPLICA**

Desired tasks : Enter **1**

6. Keep other options as default, and click on the **Create** button.

Task 6: Test the HTTPD container in ECS Cluster

1. On the left sidebar, click on the **Clusters** option present under the **Amazon ECS** section.
2. whiz ECS Cluster will be listed here, click on the **whiz**.
3. To view the ECS Instance, switch to the **Infrastructure** tab.
4. Scroll down to the Container Instances

Container instances (1) Info								
<input type="text"/> Filter container instances by property or value < 1 > ⌂								
	Container instance	Status	Type	Instance ID	Capacity...	Availability zo...	Running tasks...	Cri...
	651321a8d35a4cd...	Active	EC2	i-04ac8bbc136e...	Infra-ECS...	us-east-1d	1	76

5. **Click** on the EC2 Instance ID, and you will be redirected to running the EC2 instance.

Instances (1) Info								
<input type="text"/> Find instance by attribute or tag (case-sensitive) < 1 > ⌂								
<input type="button" value="Clear filters"/>								
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
	-	i-06ed759c4c1160881	Running	t2.micro	2/2 checks passed	No alarms	+	us-east-1a ec2-3-95-162-15

6. **Select** the EC2 Instance and switch to **Security** tab and click on security group

Instance: i-06ed759c4c1160881

Screenshot of the AWS Lambda instance details page. The Security tab is selected. The 'Security groups' section is highlighted with a red box, showing the default security group 'sg-0f8d61fe9ee5d080e'.

7. You will be redirected to the security group page, now click on **Edit Inbound rules** button

8. Click on **Add rule** button

Type : Select **SSH**

Source : Select Anywhere IPv4

9. Click on **Save Rules** button.

Screenshot of the AWS Security Group inbound rules page. The 'Type' dropdown is set to 'SSH' and the 'Source' dropdown is set to 'Anywhere'. The 'Save rules' button is highlighted with a red box.

Task 7 : SSH into the underlying EC2 instance and run Docker commands

1. Please follow the steps in [SSH into EC2 Instance](#)
2. Get the root access using the following command:

```
sudo su
```

3. Now run the updates using the following command:

```
yum -y update
```

4. Check the Docker version by running the following command:

```
docker version
```

5. Check all the docker processes running in the ECS Cluster

```
docker ps
```

```
[abhitnc@Abhishek-MacBook-Air Desktop % ssh -i "WhizKeyPair.pem" ec2-user@ec2-34-201-35-201.compute-1.amazonaws.com
Last login: Wed Jan 20 09:29:44 2021 from 49.128.166.42
 _ _|_ _|_ _|
 _| | ( \_ \_) Amazon Linux 2 (ECS Optimized)
 ____\_\_|\_\_/_\_
For documentation, visit http://aws.amazon.com/documentation/ecs
2 package(s) needed for security, out of 5 available
Run "sudo yum update" to apply all updates.
-bash: warning: setlocale: LC_CTYPE: cannot change locale (UTF-8): No such file or directory
[ec2-user@ip-172-31-1-178 ~]$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS
[ PORTS NAMES
e2e55d93d0f1        httpd:2.4          "httpd-foreground"   8 minutes ago      Up 8 minutes
  0.0.0.0:8080->80/tcp   ecs-ecs-demo-1-httpd-ee89aaf9e3ecf6b3b701
f2e5f20113e7        amazon/amazon-ecs-agent:latest  "/agent"
                           ecs-agent
[ec2-user@ip-172-31-1-178 ~]$
```

- Default ECS agent and httpd container is running in the underlying EC2 instance

Do you know ?

One unique fact about AWS ECS is its support for the FireLens container logging system. FireLens allows you to route logs from containers running on ECS to various

AWS services, such as Amazon CloudWatch Logs, Amazon Kinesis Data Firehose, or even third-party log aggregators.