NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Coaching - ECS

Cloud Infrastructure Engineering

Nanyang Technological University & Skills Union - 2022/2023

ECS Overview

Amazon Elastic Container Service (Amazon ECS) is a fully managed container orchestration service that simplifies your deployment, management, and scaling of containerized applications.

Use Cases

Modernize applications

Empower developers to build and deploy applications with enhanced security features in a fast, standardized, compliant, and cost-efficient manner with Amazon ECS.

Automatically scale web applications

Automatically scale and run web applications in multiple Availability Zones with the performance, scale, reliability, and availability of AWS.

Support batch processing

Plan, schedule, and run batch computing workloads across AWS services, including Amazon Elastic Compute Cloud (EC2), AWS Fargate, and Amazon EC2 Spot Instances.

Train NLP and AI/ML models

Train natural language processing (NLP) and other artificial intelligence (AI) / machine learning (ML) models without managing the infrastructure by using Amazon ECS with AWS Fargate.

Amazon Elastic Container Service Layers

Provisioning

Amazon Web Services Command Line Interface | Copilot | Management console | Amazon Web Services Cloud Developer Kit | Amazon Web Services Software Developer Kit

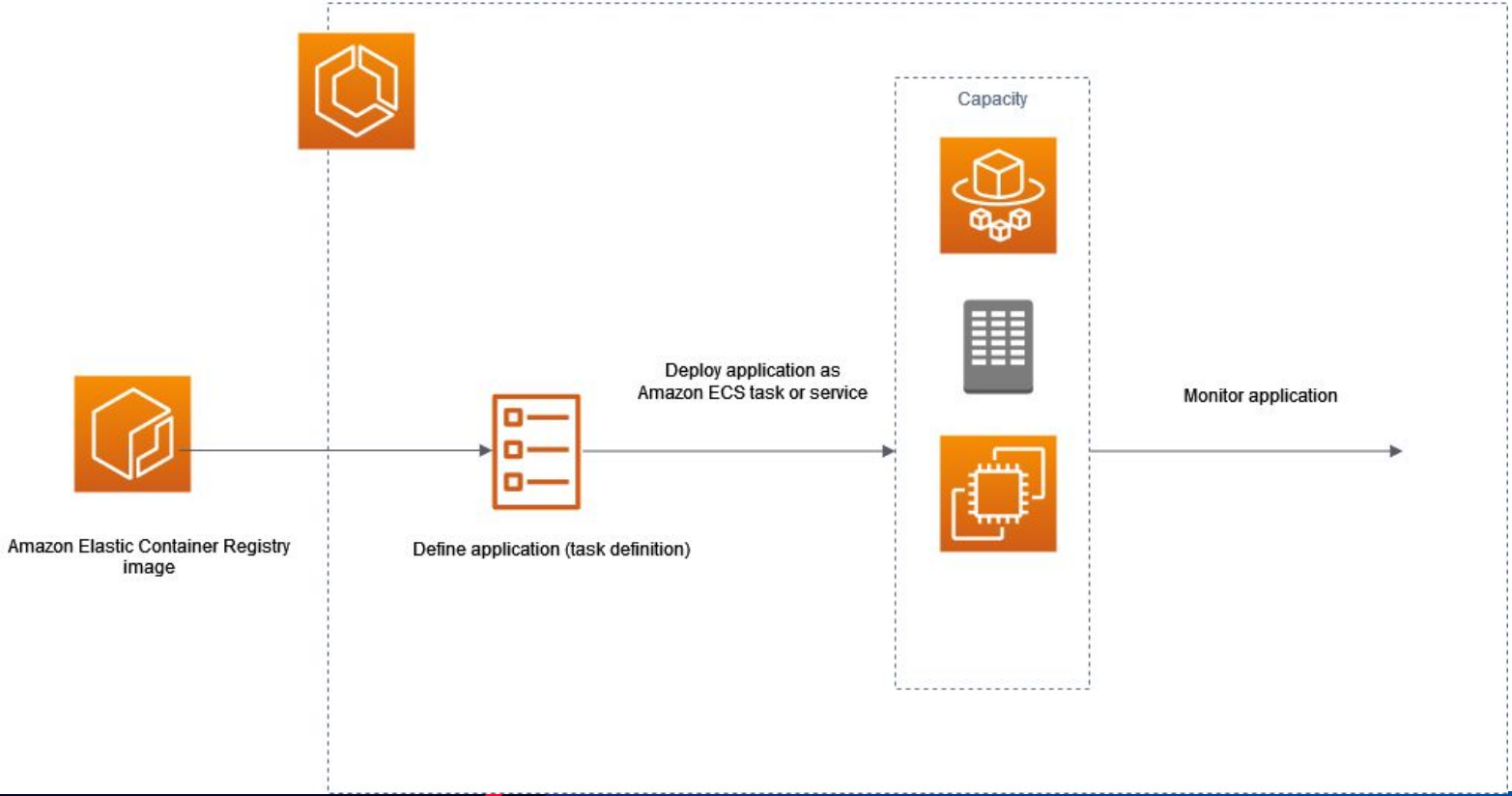
Controller

Amazon ECS scheduler

Capacity options

Amazon EC2 instances | Amazon Web Services Fargate | On-premises compute

Amazon ECS Application Lifecycle



ECS Terminologies

ECS Cluster

An Amazon ECS cluster is a logical grouping of tasks or services. In addition to tasks and services, a cluster consists of the following resources:

- The infrastructure capacity which can be any of the following:
  - Amazon EC2 instances in the AWS cloud
  - Serverless (AWS Fargate (Fargate)) in the AWS cloud
  - On-premises virtual machines (VM) or servers

ECS Capacity Provider

Fargate: With Amazon ECS on AWS Fargate capacity providers, you can use both Fargate and Fargate Spot capacity with your Amazon ECS tasks. With Fargate Spot, you can run interruption tolerant Amazon ECS tasks at a rate that's discounted compared to the Fargate price. Fargate Spot runs tasks on spare compute capacity. When AWS needs the capacity back, your tasks are interrupted with a two-minute warning.

EC2: With AWS ECS on EC2 capacity providers, you can specify the auto scaling group and the setting required for the autoscaling group.

AWS ECS Task Definition

A *task definition* is a blueprint for your application. It is a text file in JSON format that describes the parameters and one or more containers that form your application.

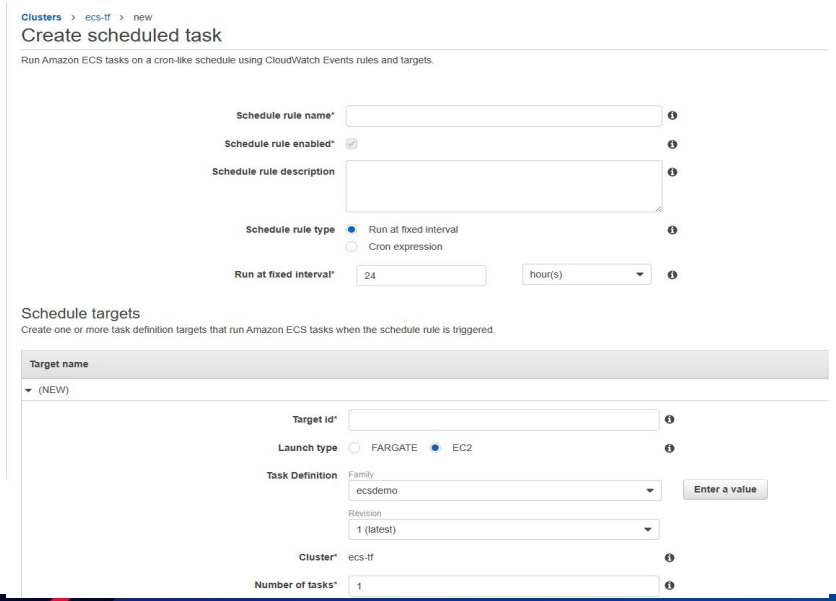
The following are some of the parameters that you can specify in a task definition:

- The Docker image to use with each container in your task
- How much CPU and memory to use with each task or each container within a task
- The launch type to use, which determines the infrastructure that your tasks are hosted on
- The Docker networking mode to use for the containers in your task
- The logging configuration to use for your tasks
- Whether the task continues to run if the container finishes or fails
- The command that the container runs when it's started
- Any data volumes that are used with the containers in the task
- The IAM role that your tasks use

ECS Service vs Tasks

Service : Typically used for long-running tasks / containers that can be stopped and restarted. E.g. A web application

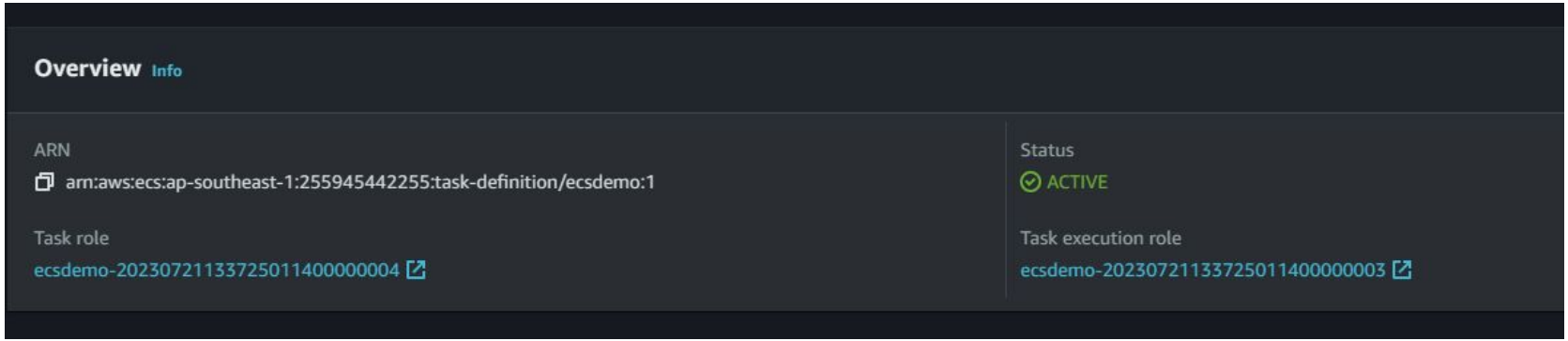
Task: A standalone task that is used to run a short running task. For e.g. A batch job, or a scheduled cron job to run at a specific time everyday and then exit



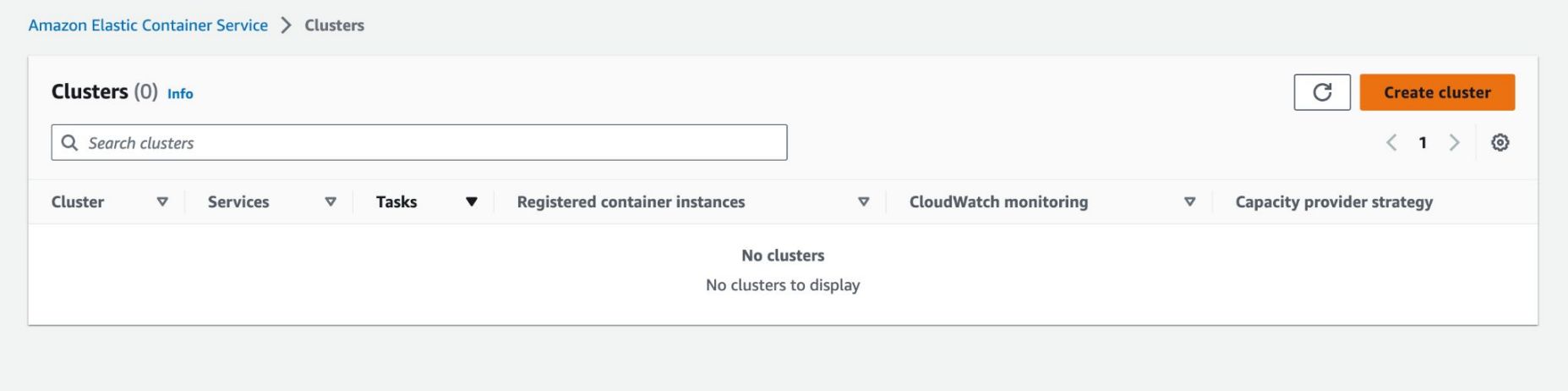
ECS TaskExecutionRole vs TaskRole

TaskExecutionRole: IAM role that executes ECS actions such as pulling the image and storing the application logs in cloudwatch.

TaskRole: IAM role used by the task itself. For example, if your container wants to call other AWS services like S3, SQS, etc then those permissions would need to be covered by the TaskRole



Creating a ECS Cluster



Create cluster Info

An Amazon ECS cluster groups together tasks, and services, and allows for shared capacity and common configurations. All of your tasks, services, and capacity must belong to a cluster.

Cluster configuration

Cluster name

jazeel-cluster

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

Networking Info

By default tasks and services run in the default subnets for your default VPC. To use the non-default VPC, specify the VPC and subnets.

VPC

Use a VPC with public and private subnets. By default, VPCs are created for your AWS account. To create a new VPC, go to the VPC Console.

vpc-01eb8331fac862b2e
sandbox-vpc

Subnets

Select the subnets where your tasks run. We recommend that you use three subnets for production.

Choose subnets

subnet-0623d78431b777e3e public
public-subnet-01
ap-southeast-1a 10.0.1.0/24

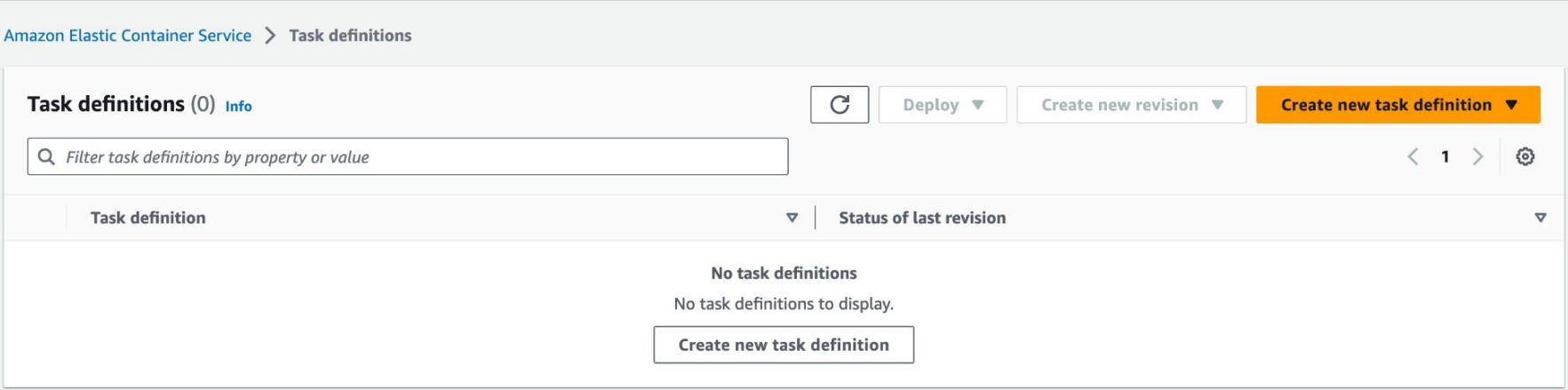
subnet-02a6bf9a87a5dec14 public
public-subnet-02
ap-southeast-1b 10.0.2.0/24

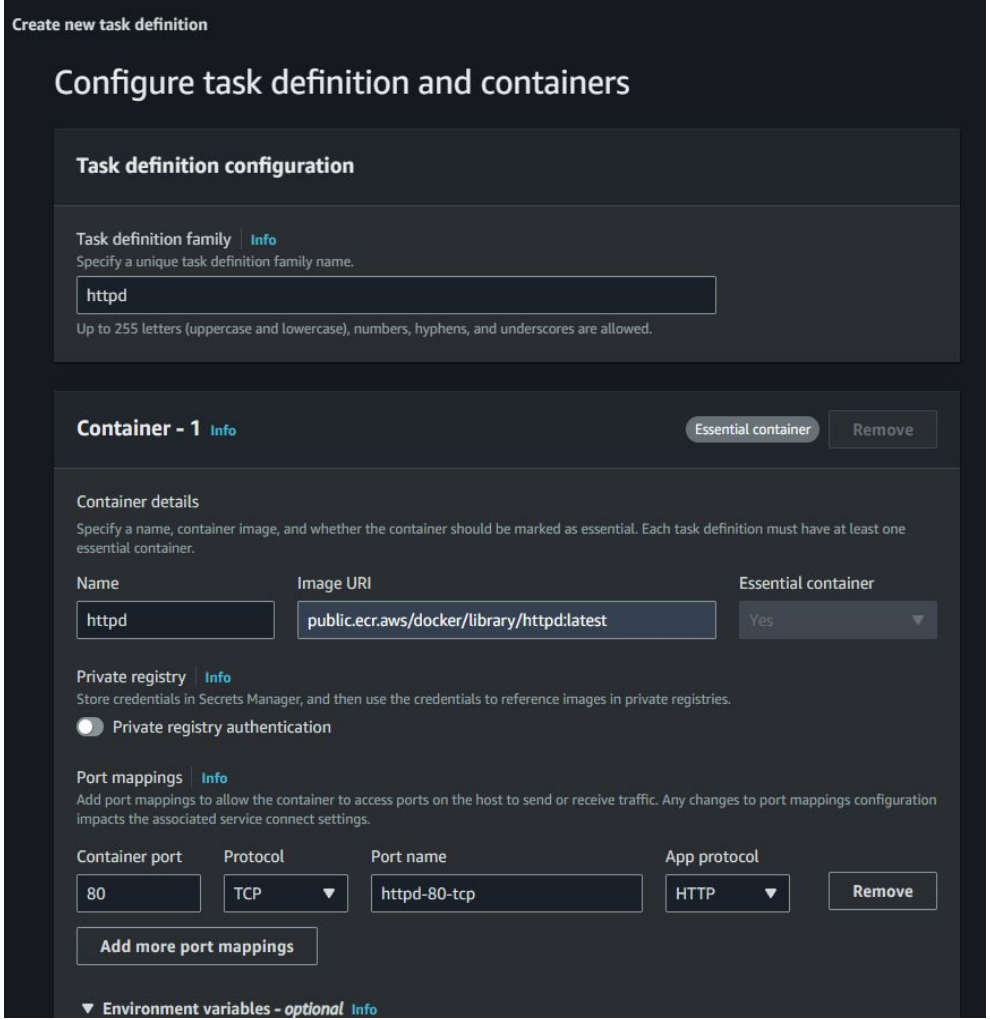
subnet-07d728c6db3bd830b public
public-subnet-03
ap-southeast-1c 10.0.3.0/24

Default namespace - *optional*

Select the namespace to specify a group of services that make up your application. You can overwrite this value at the service level.

Task Definition Creation




Create new task definition
Configure task definition and containers
Task definition configuration
Task definition family Info
Specify a unique task definition family name.
httpd
Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.
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
public.ecr.aws/docker/library/httpd:latest
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
Remove
Add more port mappings
Environment variables - optional Info

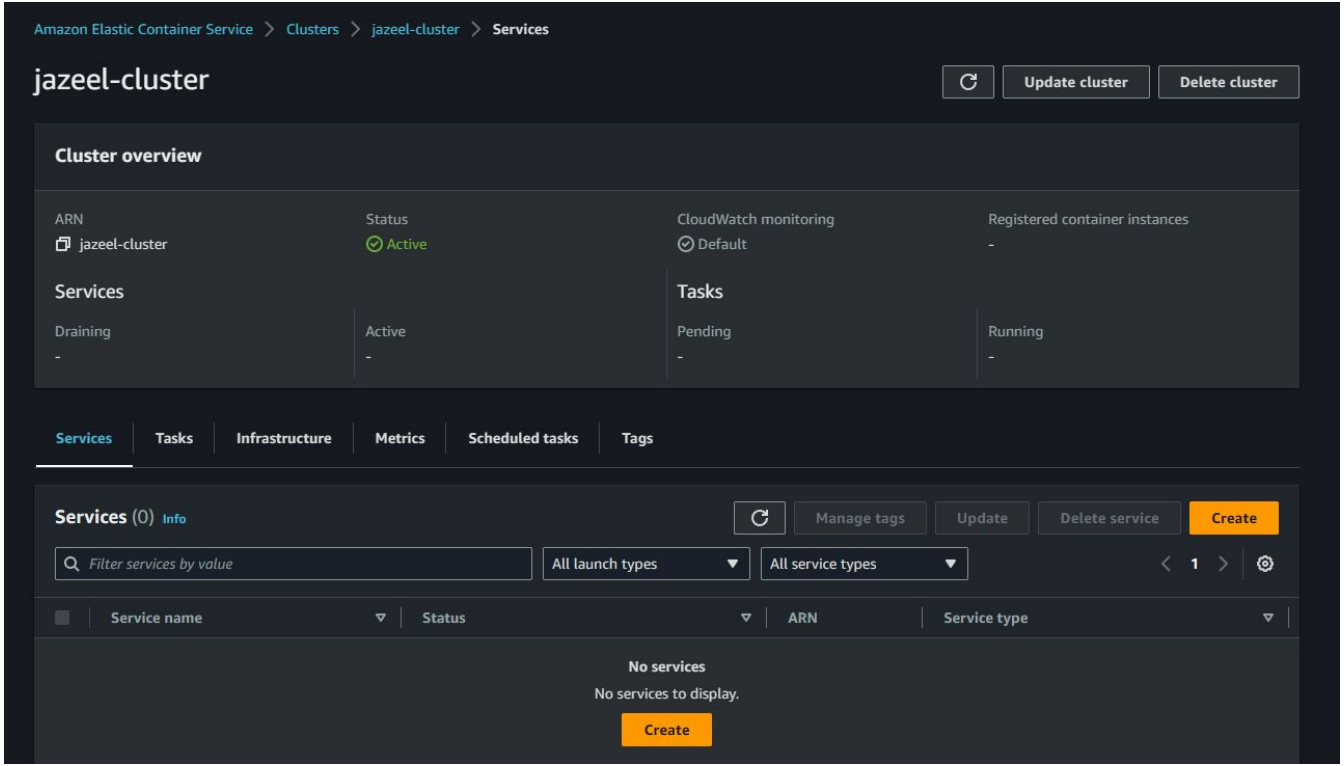
Task Definition Example

JSON

task_definition.json

```
{
    "taskDefinitionArn": "arn:aws:ecs:ap-southeast-1:255945442255:task-definition/ecsdemo:1",
    "containerDefinitions": [
        {
            "name": "ecs-sample",
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "cpu": 0,
            "portMappings": [
                {
                    "name": "ecs-sample",
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "environment": [],
            "mountPoints": [],
            "volumesFrom": [],
            "startTimeout": 30,
            "stopTimeout": 120,
            "privileged": false,
            "readonlyRootFilesystem": false,
            "interactive": false,
            "pseudoTerminal": false,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/aws/ecs/ecsdemo/ecs-sample",
                    "awslogs-region": "ap-southeast-1",
```

Service Creation



Deployment configuration

Application type | Info

Specify what type of application you want to run.

- (•) **Service** — Launch a group of tasks handling a long-running computing work that can be stopped and restarted. For example, a web application.
- ( ) **Task** — Launch a standalone task that runs and terminates. For example, a batch job.

Task definition

Select an existing task definition. To create a new task definition, go to Task definitions.

- [ ] **Specify the revision manually**
  Manually input the revision instead of choosing from the 100 most recent revisions for the selected task definition family.

Family	Revision
httpd	1 (LATEST)

Service name

Assign a unique name for this service.

httpd

Service type | Info

Specify the service type that the service scheduler will follow.

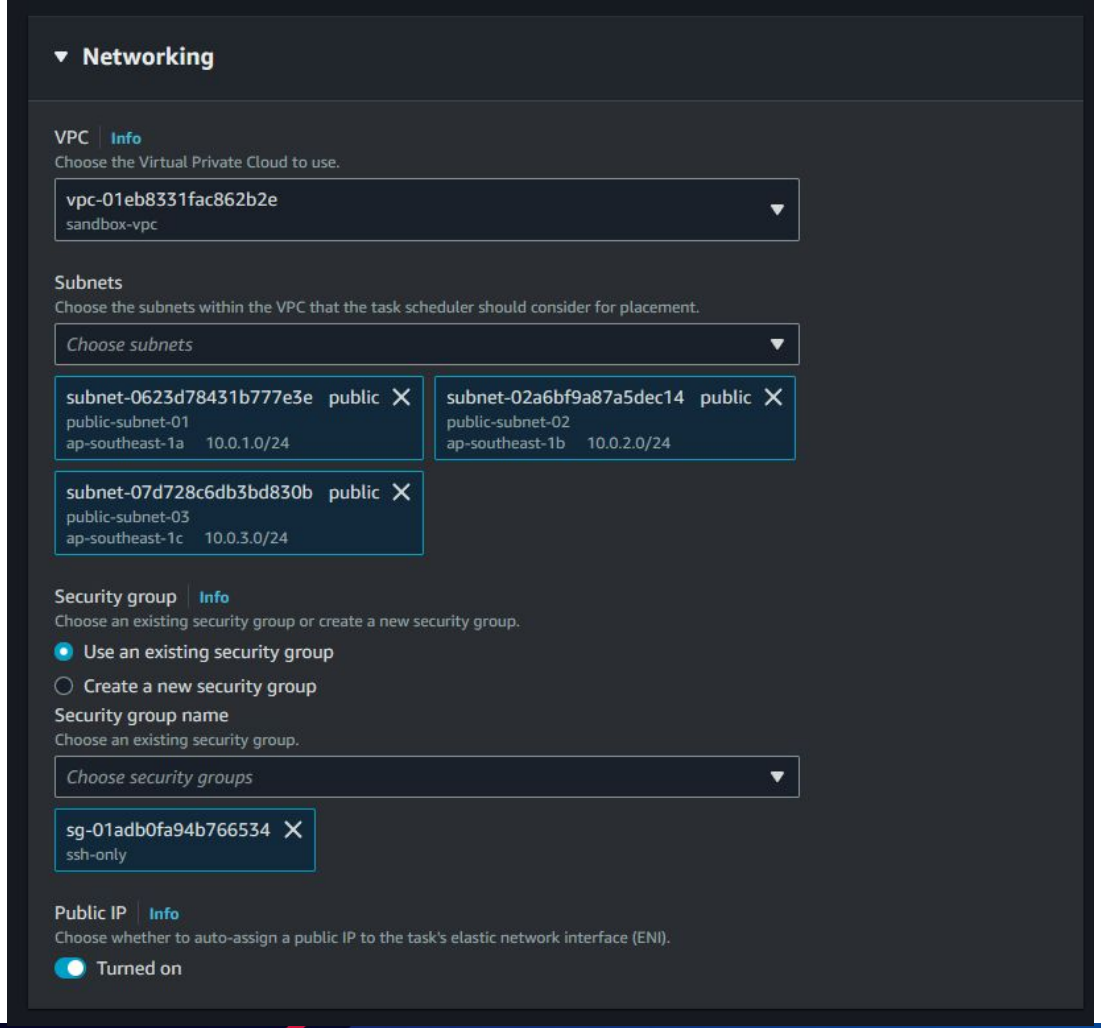
- (•) **Replica** — Place and maintain a desired number of tasks across your cluster.
- ( ) **Daemon** — Place and maintain one copy of your task on each container instance.

Desired tasks

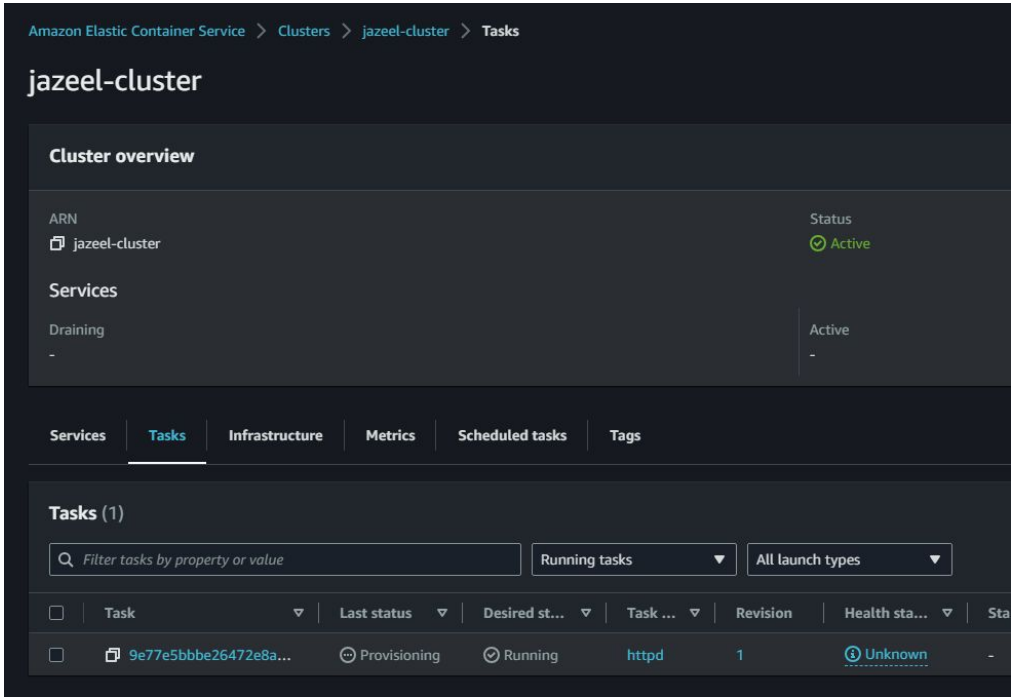
Specify the number of tasks to launch.

1

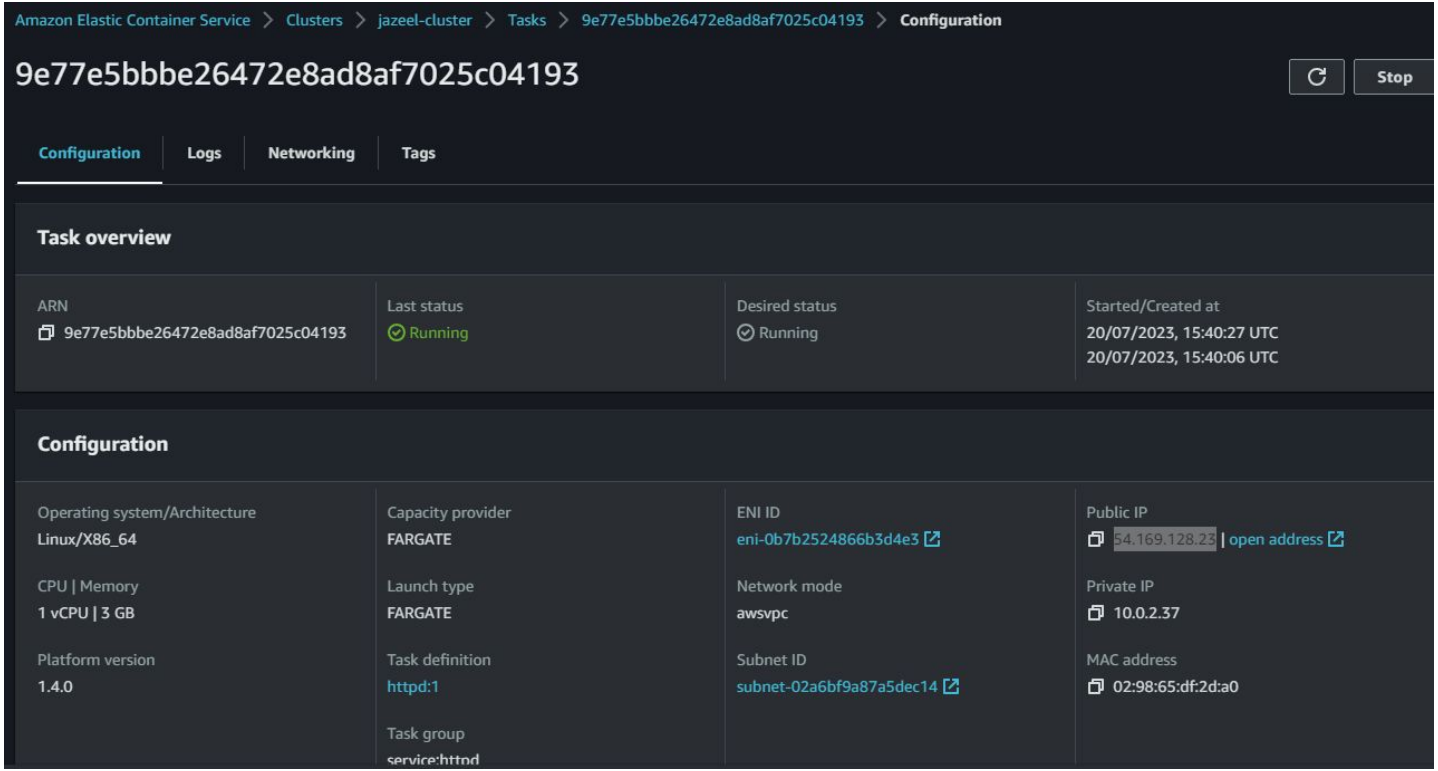
▶ **Deployment options**


Networking
VPC Info
Choose the Virtual Private Cloud to use.
vpc-01eb8331fac862b2e
sandbox-vpc
Subnets
Choose the subnets within the VPC that the task scheduler should consider for placement.
Choose subnets
subnet-0623d78431b777e3e public
public-subnet-01
ap-southeast-1a 10.0.1.0/24
subnet-02a6bf9a87a5dec14 public
public-subnet-02
ap-southeast-1b 10.0.2.0/24
subnet-07d728c6db3bd830b public
public-subnet-03
ap-southeast-1c 10.0.3.0/24
Security group Info
Choose an existing security group or create a new security group.
Use an existing security group
Create a new security group
Security group name
Choose an existing security group.
Choose security groups
sg-01adb0fa94b766534
ssh-only
Public IP Info
Choose whether to auto-assign a public IP to the task's elastic network interface (ENI).
Turned on

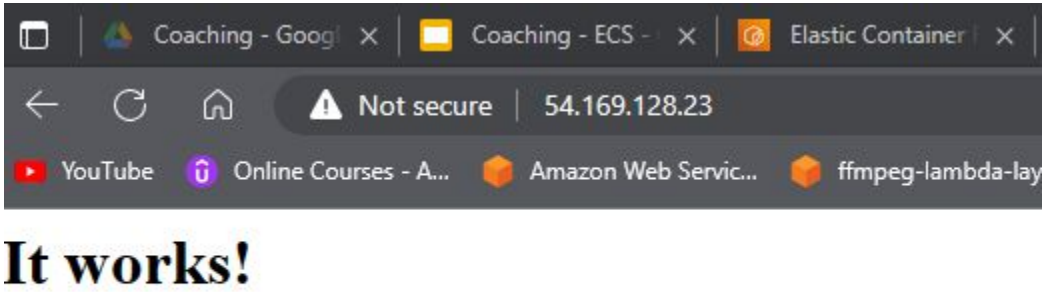
Getting IP of Task



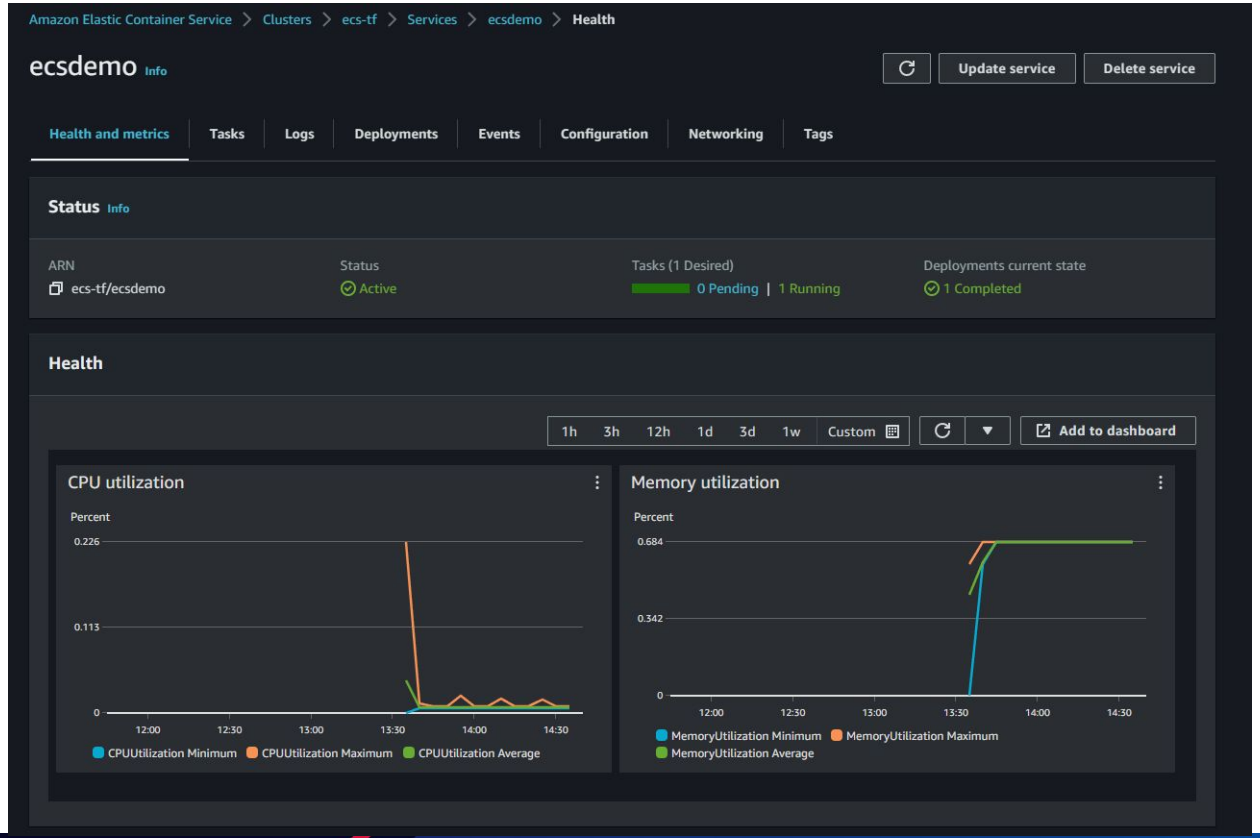
Getting IP of Task



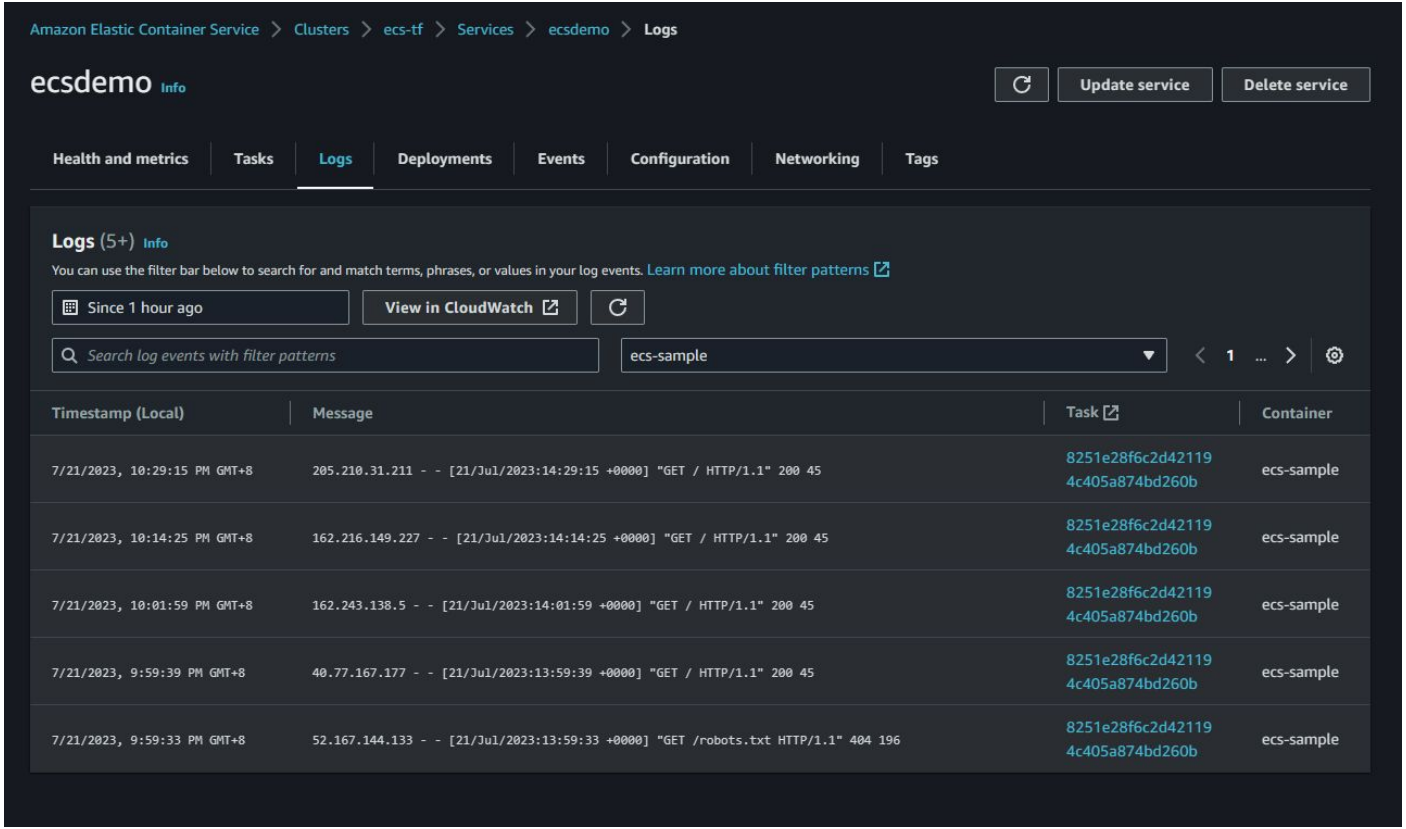
Getting IP of Task



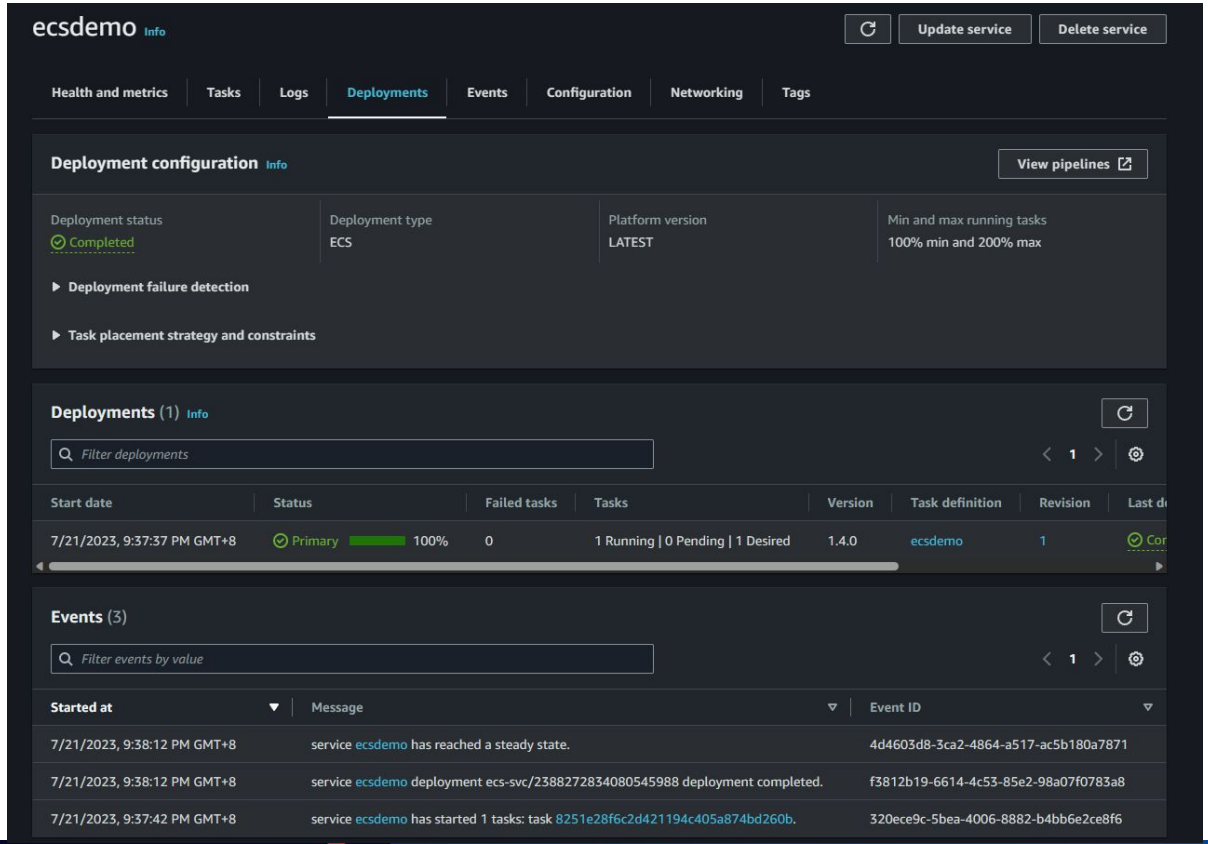
Service In-depth - Health and metrics



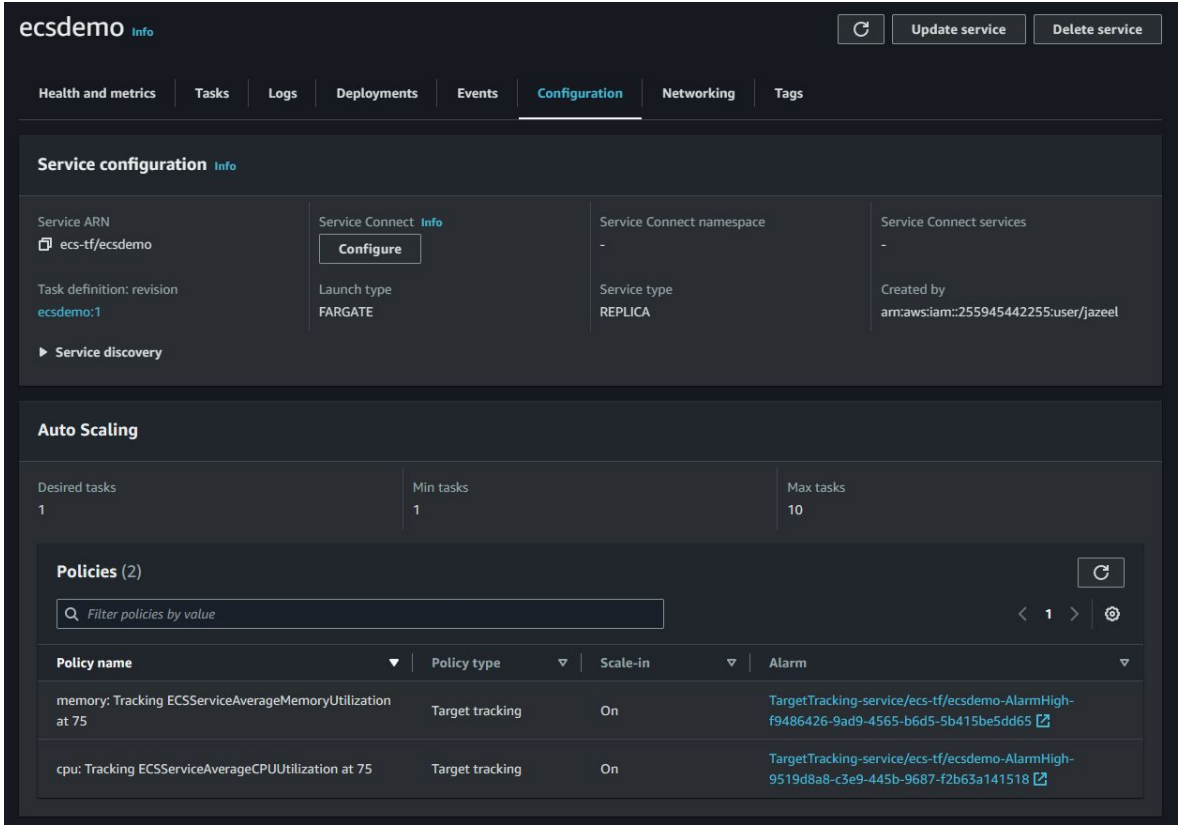
Service In-depth - Logs



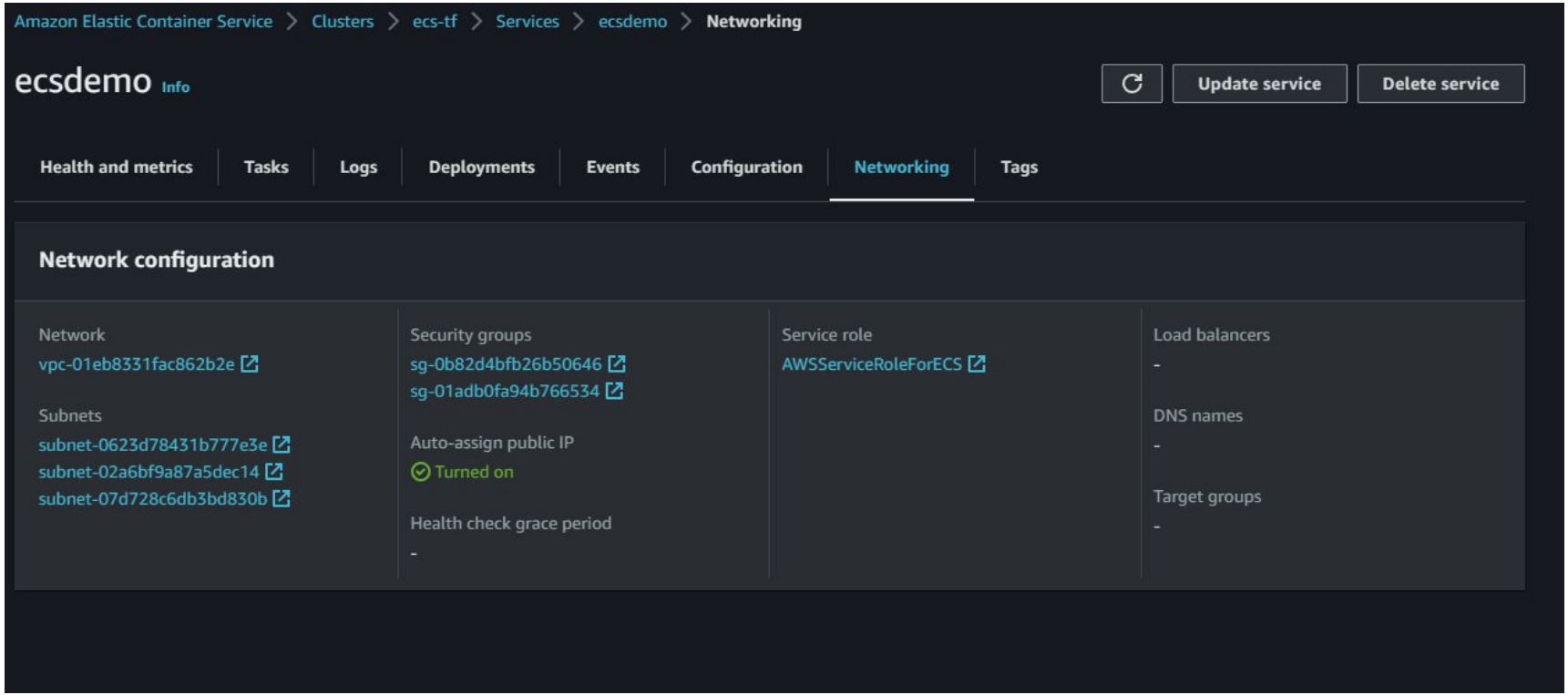
Service In-depth - Deployment



Service In-depth - Service Auto-Scaling



Service In-depth - Networking



Create ECS resources using Terraform

Terraform ECS Module

terraform-aws-modules/terraform-aws-ecs: Terraform module which creates AWS ECS resources 🇺🇦 (github.com)

Activity

Create a new repo in Github and clone it to your local computer

.gitignore template -> Terraform

Add README.md

Activity

Create a main.tf file with below content:

Change the variables in the lines which has # Change in the comment. (line 18 and 29)

https://github.com/jaezeu/hello-node/blob/main/terraform/main.tf

Activity

Create a backend.tf file with below content:

```
terraform {
  backend "s3" {
    bucket = "sctp-ce2-tfstate-bkt"
    key    = "jazeel-ecs-1.tfstate"   #Change the value  of this to yourname-ecs-1.tfstate for  example
    region = "ap-southeast-1"
  }
}
```

Activity

Create a provider.tf file with below content:

```
provider "aws" {
  region = "ap-southeast-1"
}
```

Activity

Once the files above have been created, Run the following commands:

terraform init

terraform plan

terraform apply

Activity

1) Access the cluster you have created
2) Go to the tasks(Slide 13)
3) Get public IP and display in browser(Slide 14)

CI/CD Workflow