Containers on AWS

AWS KRUG 컨테이너 소모임

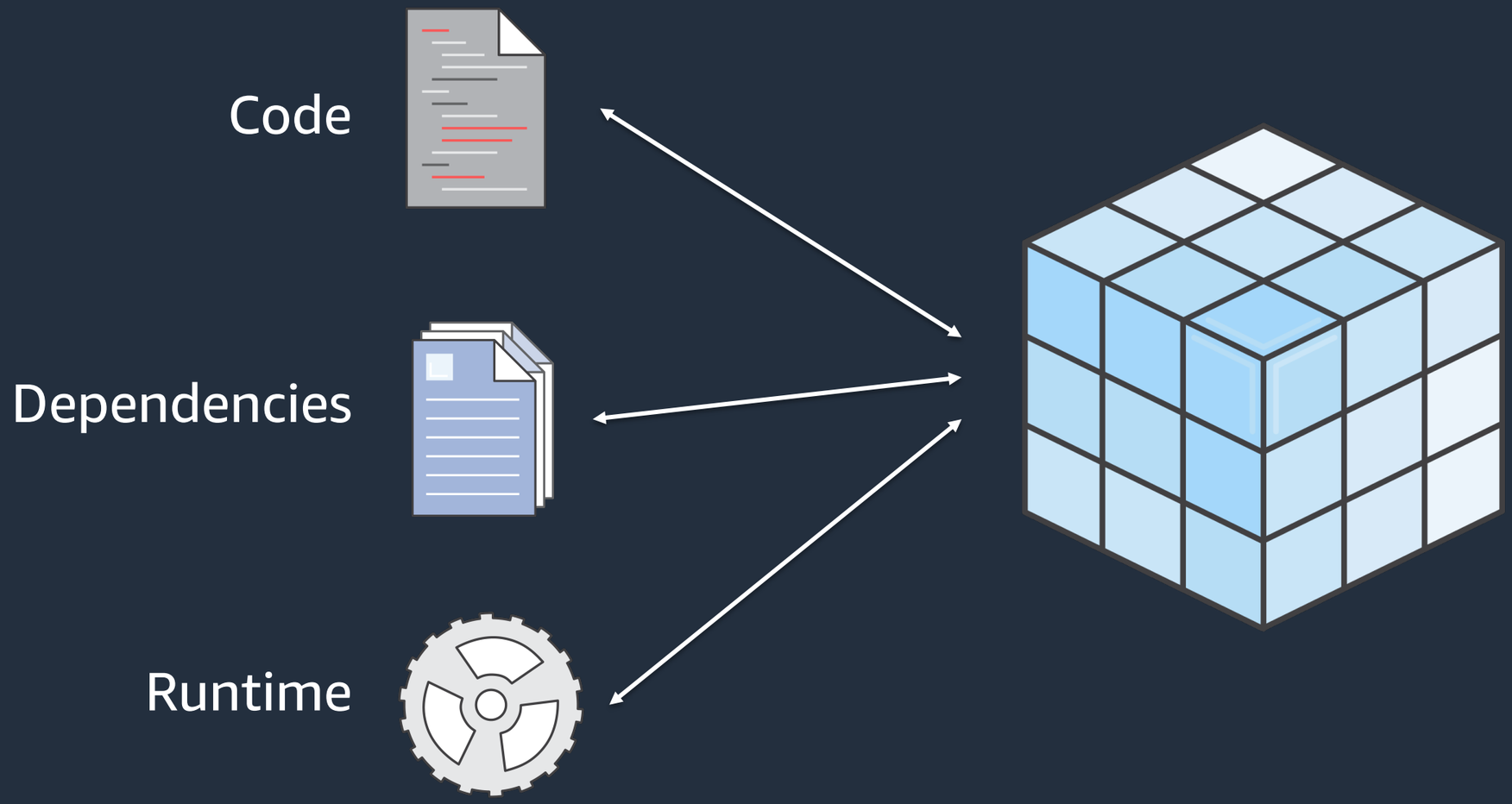
김지민 솔루션즈 아키텍트, AWS

Agenda

1. 컨테이너가 대세인 이유?
2. (너무) 다양한 AWS 컨테이너 서비스들
3. Amazon EKS, ECS, Fargate
4. 컨테이너 애플리케이션 운영하기 – 로깅, 모니터링
5. 쉽게 컨테이너 시작하기

aws

컨테이너란?



하나의 컨테이너로 시작

aws

컨테이너가 아주 많아지게 된다면?

aws

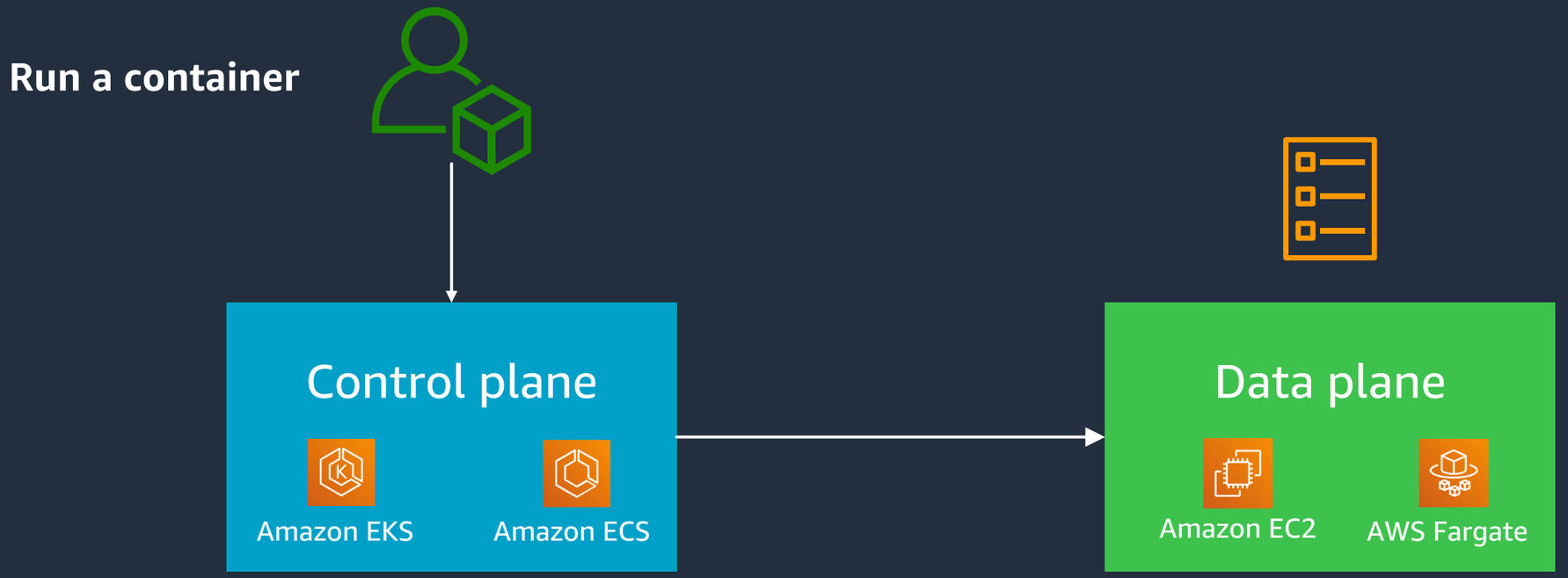
컨테이너 오케스트레이터

Control plane

Data plane

aws

컨테이너 오케스트레이터



사용자가 원하는 상태(**Desired State**)로 동작하도록 관리(**Schedule**)하는 것

AWS container services landscape

APPLICATION NETWORKING

Service discovery and service mesh

AWS Cloud Map

AWS App Mesh

MANAGEMENT

Deployment, scheduling, scaling, and management of containerized applications

Amazon Elastic Container Service (Amazon ECS)

Amazon Elastic Kubernetes Service (Amazon EKS)

HOSTING

Where the containers run

Amazon Elastic Compute Cloud (Amazon EC2)

AWS Fargate

IMAGE REGISTRY

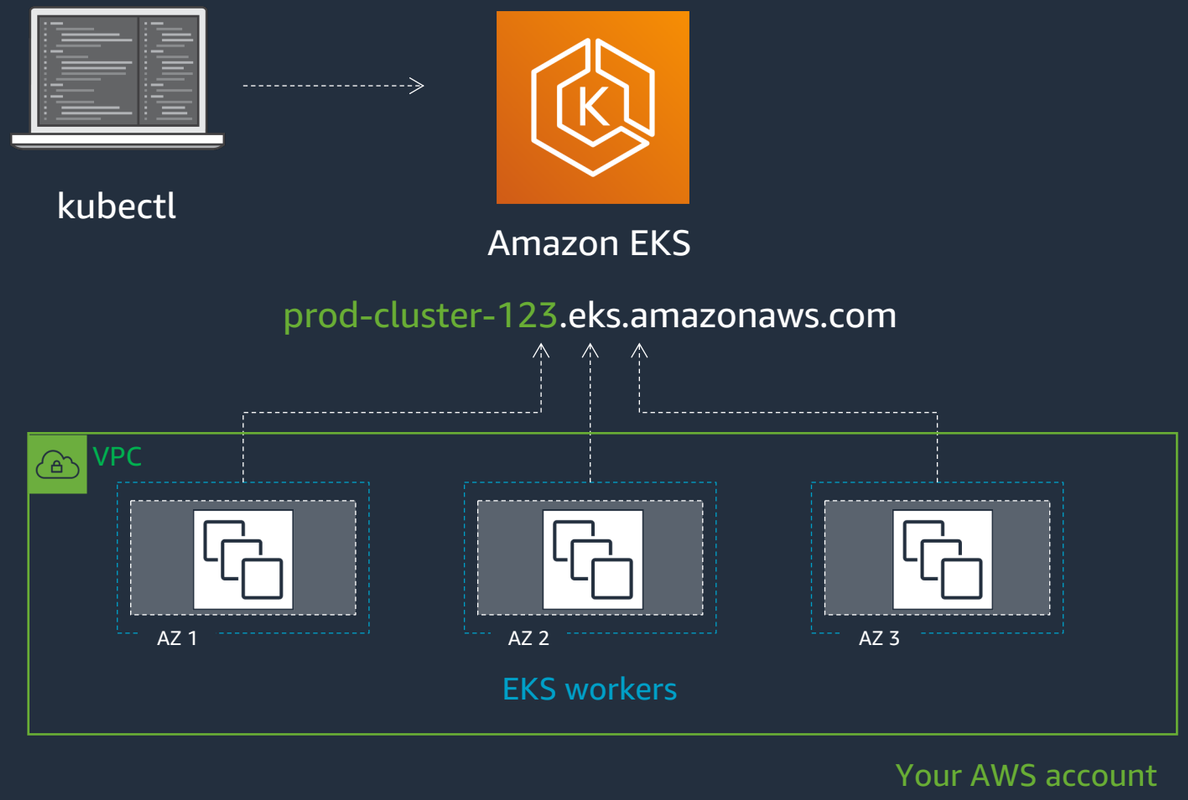
Container image repository

Amazon Elastic Container Registry (Amazon ECR)

Amazon EKS

aws

Amazon EKS Architecture



Amazon EKS 특징

Security First

Built for Production

Native and upstream

Seamless Cloud integrations

Committed to Open Source

Amazon ECS

aws

Amazon ECS 구성 요소

작업이 실행되는 논리적 그룹

실제 컨테이너 작업(최소 실행 단위)

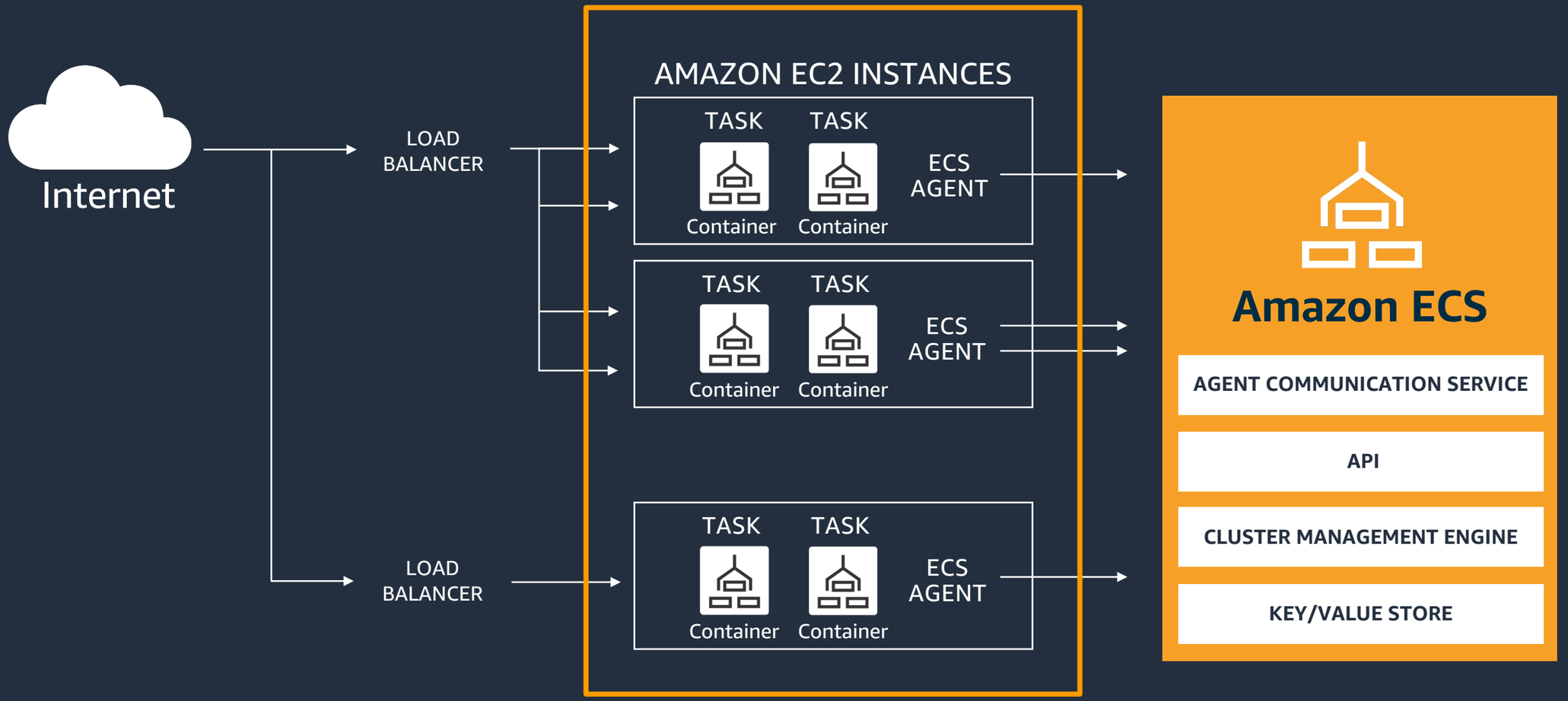
하나 혹은 둘 이상의 컨테이너의 묶음

작업을 지속적으로 관리

실패한 작업을 자동으로 대체(작업 수 유지)

aws

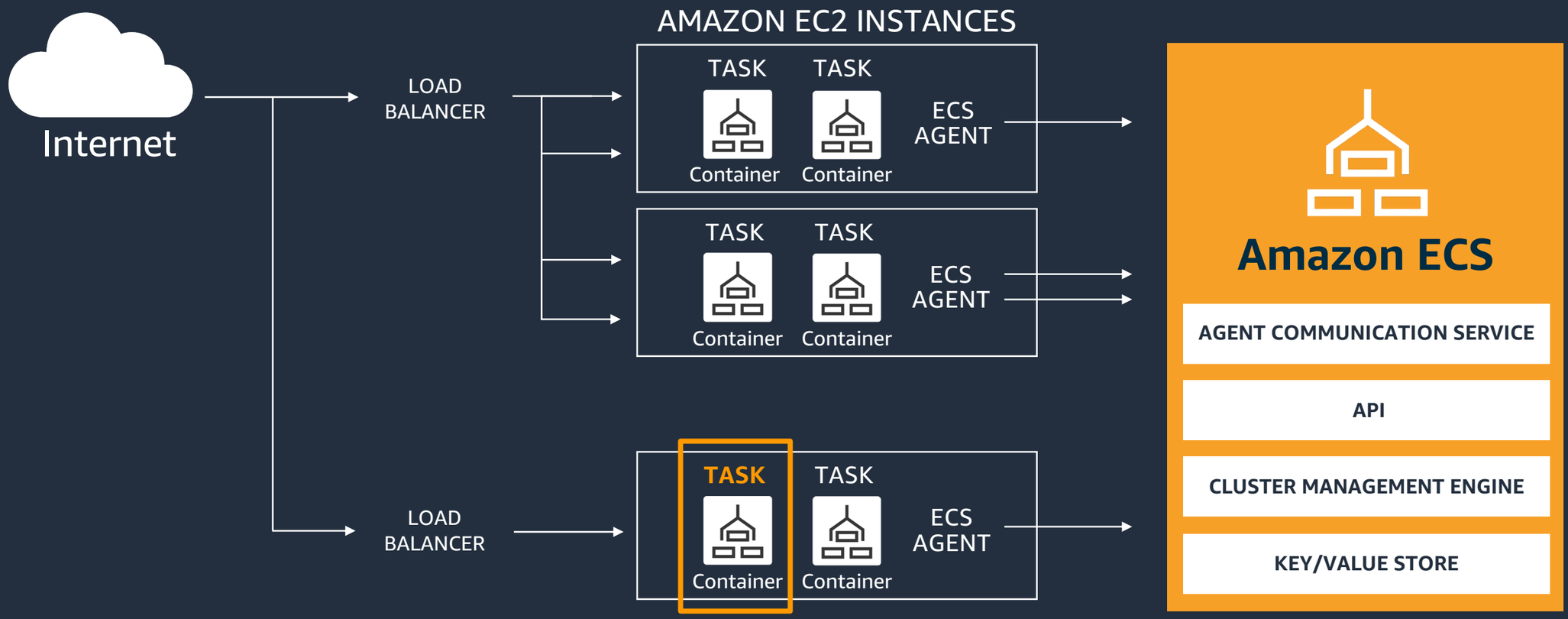
Amazon ECS



aws

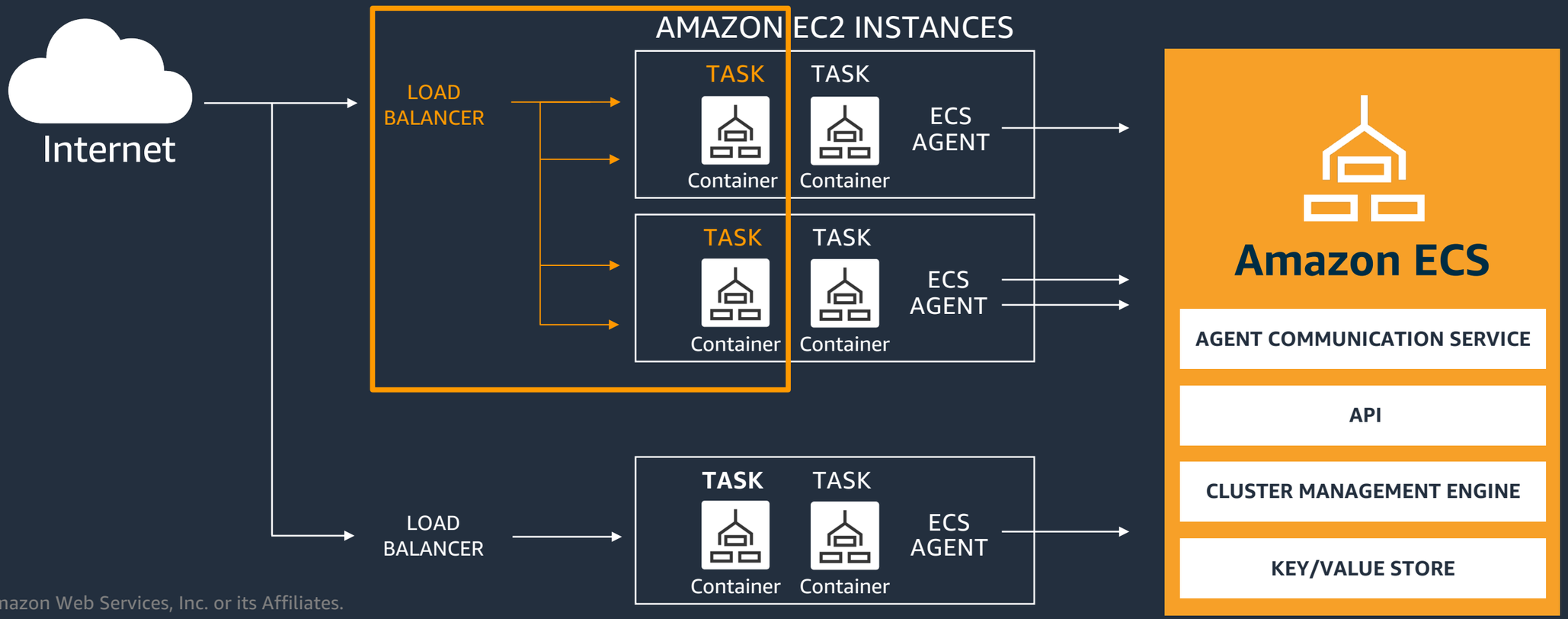
Amazon ECS - 작업

- ECS에서 **최소 실행 단위**
- 작업 정의 내용을 기반으로 ECS 클러스터에 속하는 인스턴스 및 Fargate 배포
- 한 작업 당 최대 10개 컨테이너 가능, 모두 같은 호스트에 배치



Amazon ECS - 서비스

- 설정한 작업 수를 자동 유지 및 복구
- 서비스 유형(replica, daemon), 작업 배치 전략 선택 가능
- 배포 유형: 롤링 업데이트(ECS로 제어), 블루/그린 배포(CodeDeploy로 제어)
- ALB를 통해 효율적으로 부하 분산
- 서비스 Auto Scaling 적용 가능



작업 정의 (Task Def)

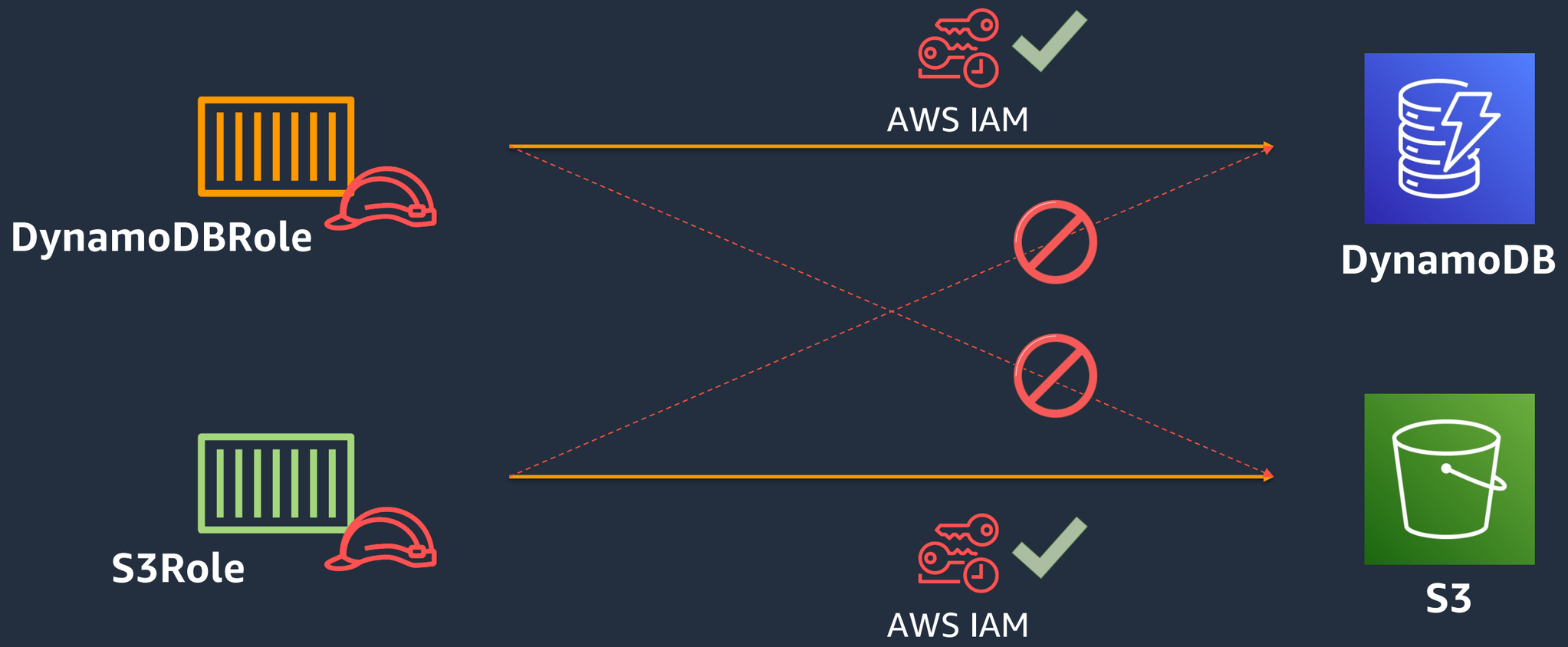
- ECS에서 컨테이너를 실행하려면 작업 정의가 필요
- 컨테이너 이미지 맵핑을 통한 컨테이너 정의
- 배포 타입 설정 가능: Fargate, EC2
- 작업 역할을 부여해 API 요청을 받을 때 권한에 따라 동작 가능
- 작업 크기 설정 가능
- App Mesh, FireLens 등과 통합 활성화 가능
- 다양한 볼륨 선택권 제공

aws

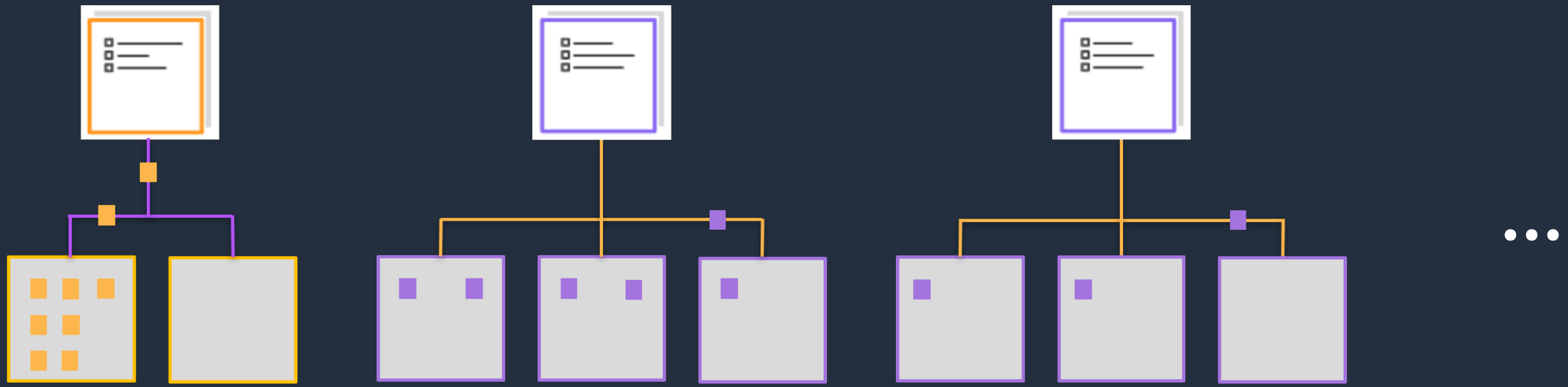
작업 정의 (Task Def)

```
{
  "containerDefinitions": [
    {
      "name": ”myApp",
      "image": ”myApp:v12",
      "essential": true,
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80
        }
      ],
      "readonlyRootFilesystem": true,
      "memory": 500,
      "cpu": 100,
...
    "logConfiguration": {
      "logDriver": "awslogs",
      "options": {
        "awslogs-group": "ecs-log-streaming",
        "awslogs-region": "us-west-2"
      }
    }
  }
],
  "family": "hello_world”,
  "taskRoleArn”:"arn:aws:iam::123456789012:role/MyTaskRole"
}
```

ECS 작업에 부여하는 IAM 역할



다양한 작업 배치 전략



Binpacking

Spread

One Task per Host

...

aws

AWS Fargate

aws

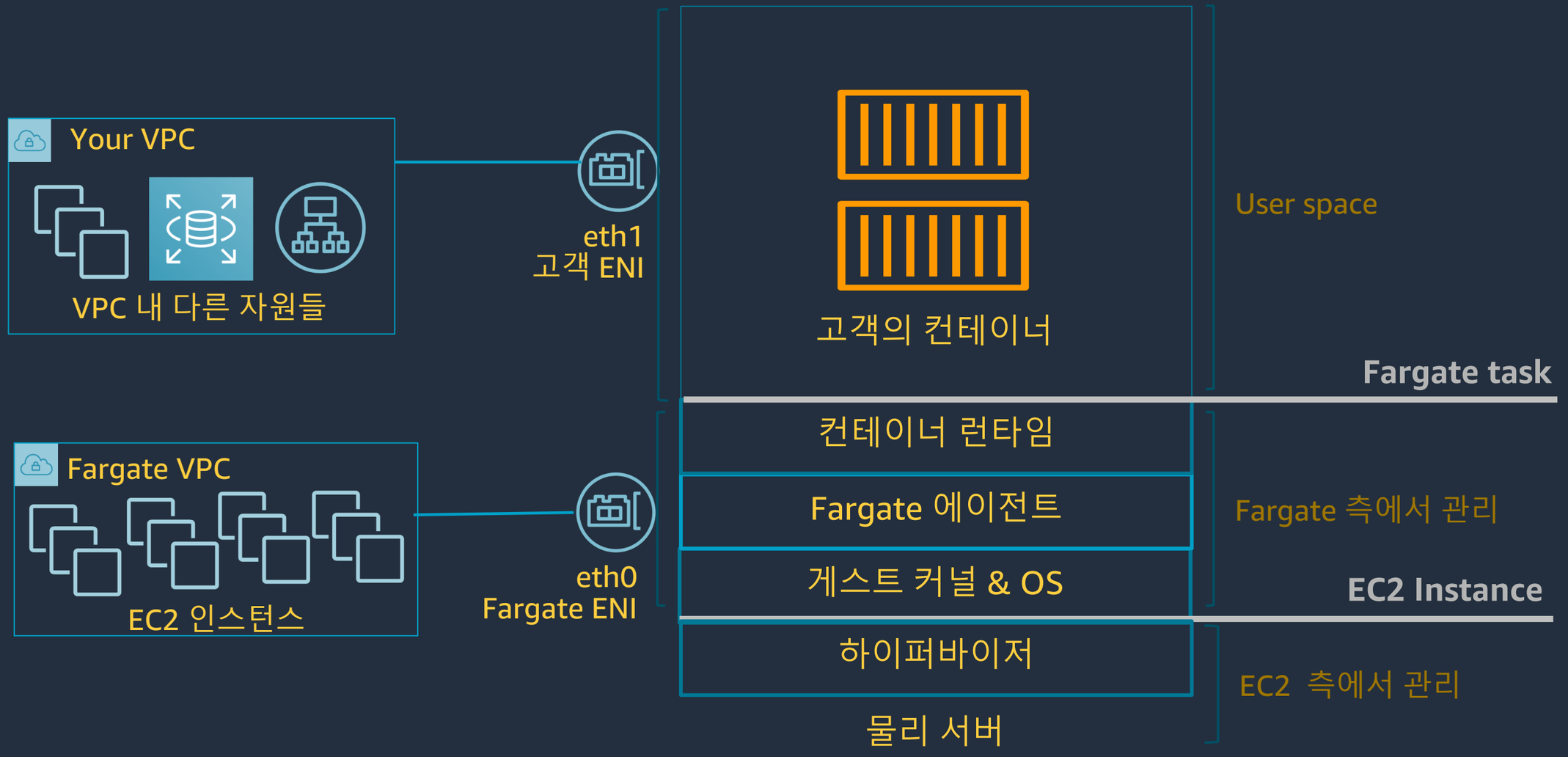
Fargate?

AWS Fargate

고객들이 기반 VM을 관리할 필요 없이
컨테이너를 사용할 수 있게 하는 컴퓨팅 엔진

aws

Fargate Data Plane



aws

Fargate의 장점

관리 업무로 인한 오버헤드 경감

까다로운 컨테이너 클러스터 관리를 AWS에 위임함으로써 고객은 애플리케이션에만 집중

기존 컨테이너 그대로 배포 가능

기존의 컨테이너 변경 불필요

현재 쿠버네티스, ECS 클러스터의 서비스, 워크플로우 그대로 이용

필요한 만큼만 & 쉬운 연동

컨테이너 실행에 필요한 자원 만큼만 비용 지불

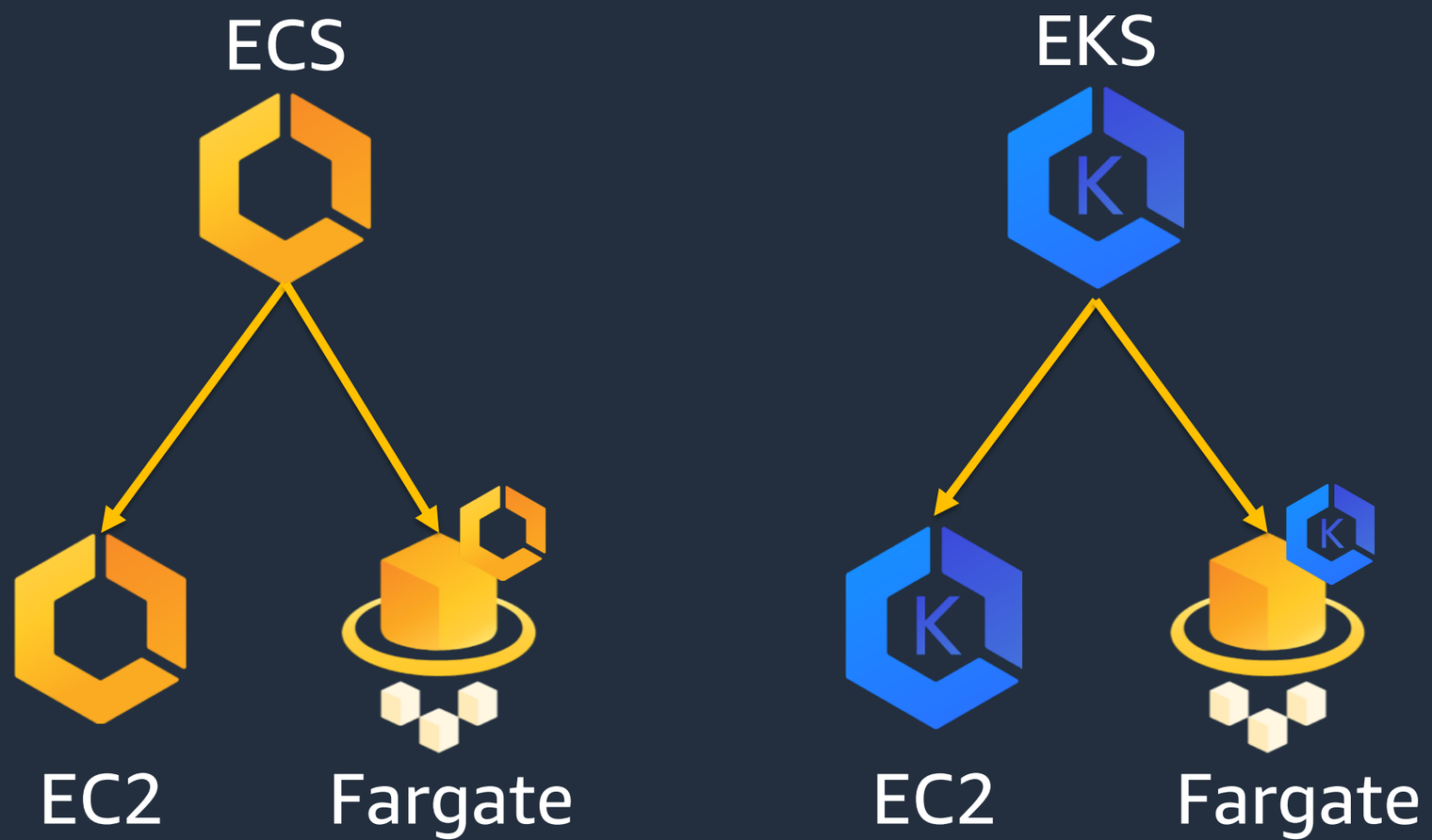
기존 AWS 네트워크, 보안과 네이티브하게 통합하여 사용

https://aws.amazon.com/blogs/containers/under-the-hood-fargate-data-plane/

필요에 따른 조합

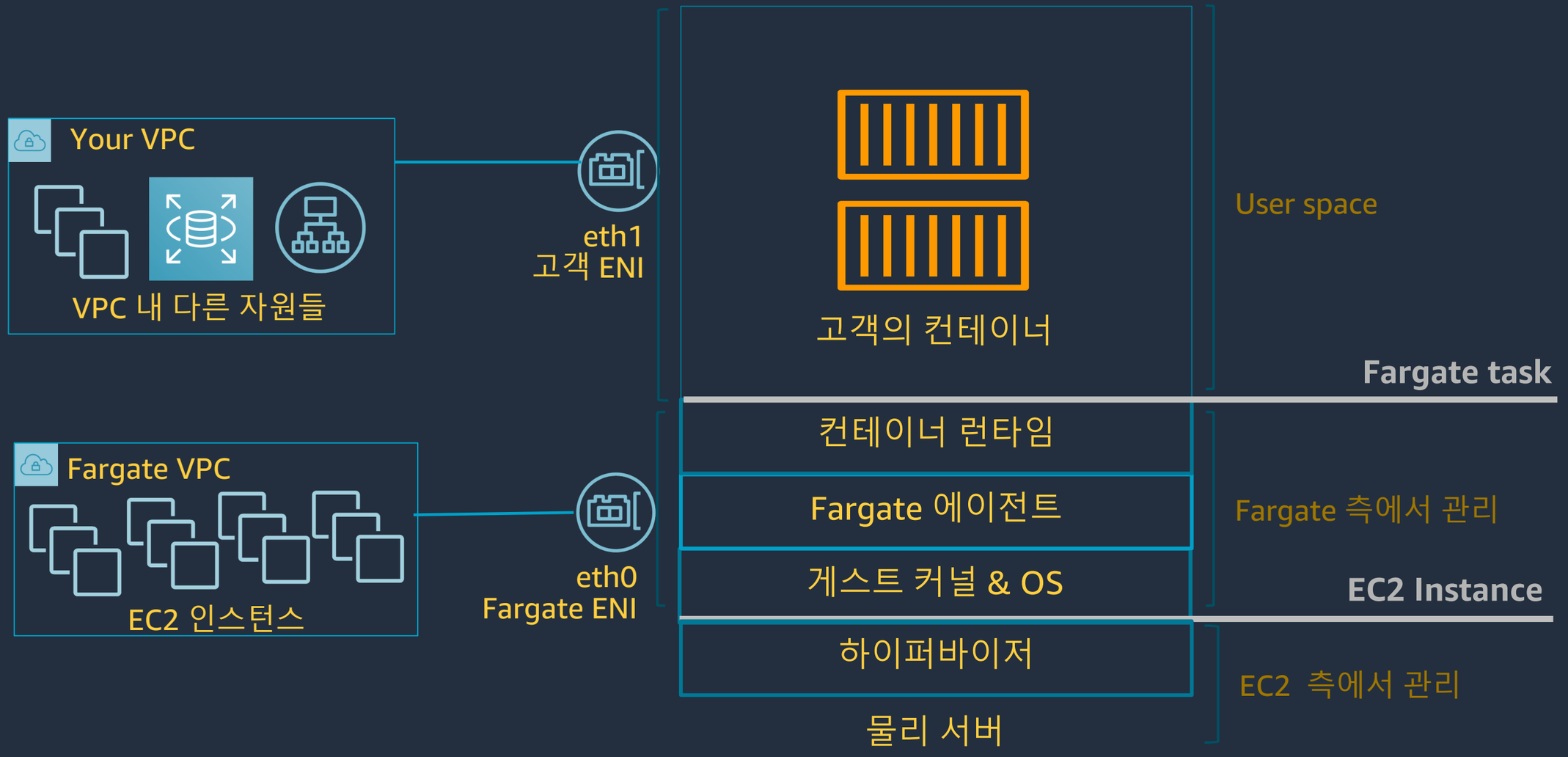
오케스트레이션

컴퓨트 엔진



aws

Fargate Data Plane



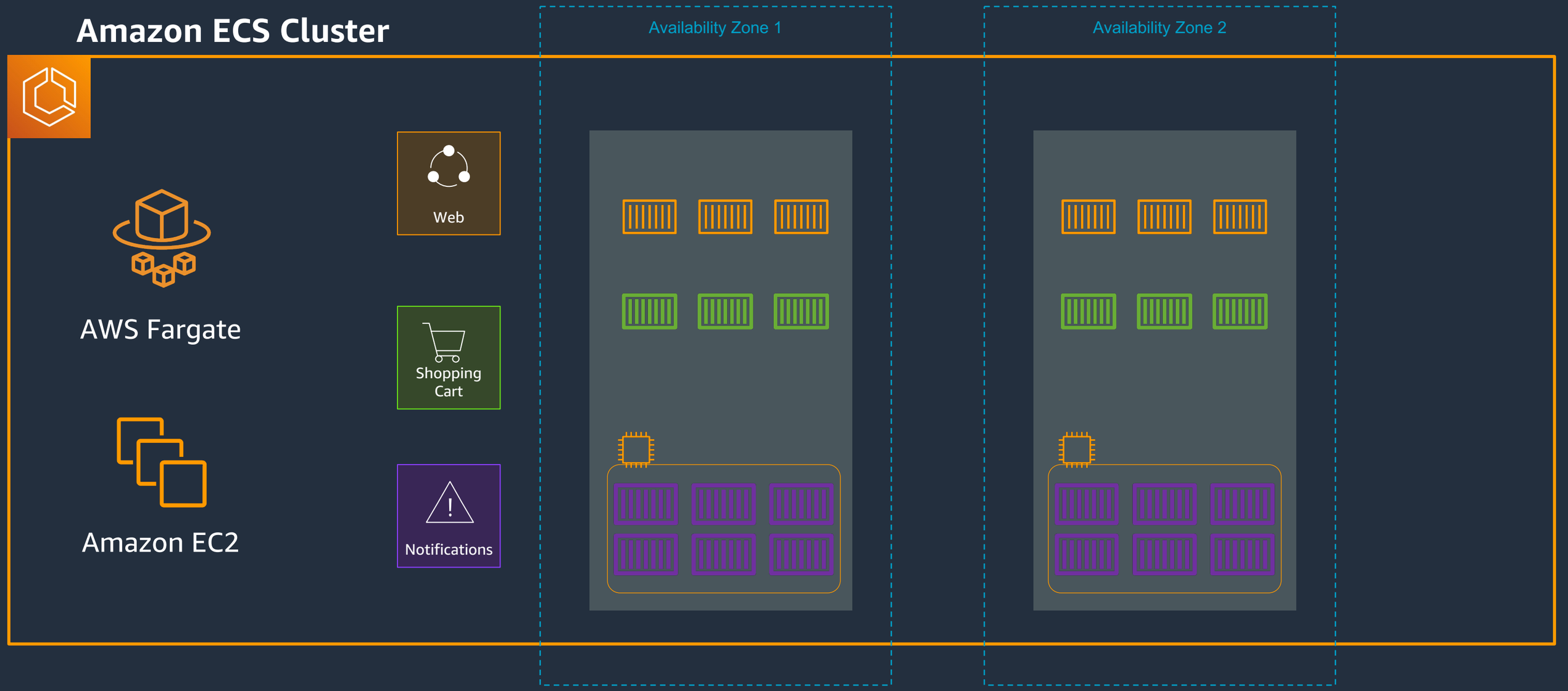
aws

CPU와 메모리 용량 범위

CPU	Memory
256 (.25 vCPU)	512MB, 1GB, 2GB
512 (.5 vCPU)	1GB to 4GB
1024 (1 vCPU)	2GB to 8GB
2048 (2 vCPU)	4GB to 16GB
4096 (4 vCPU)	8GB to 30GB

aws

Amazon EC2와 Fargate의 Hybrid Architecture



https://ecs-cats-dogs.workshop.aws/ko/01_intro.html

aws

AWS 컨테이너 서비스들

Amazon Elastic Container Registry (ECR)

- 완전 관리형 **컨테이너 레지스트리**
- 배포 워크플로우 간소화

  - Amazon EKS, Amazon ECS, Amazon Lambda, AWS Fargate
- 권한 제어 및 수명 주기 정책 규칙 설정
- 이미지 취약성 스캐닝 기능
- 퍼블릭/프라이빗 레파지토리 및 퍼블릭 갤러리
- 교차 리전/교차 계정 복제

https://ecs-cats-dogs.workshop.aws/ko/ecr.html

Amazon CloudWatch

Amazon CloudWatch를 사용하여 Amazon ECS 리소스를 모니터링

Amazon CloudWatch는 Amazon ECS에서 데이터를 수집하고 처리하여 실시간에 가까운 메트릭으로 처리

Amazon CloudWatch Alarms

Amazon CloudWatch Logs

Amazon CloudWatch Events

Amazon CloudTrail Logs

aws

Amazon CloudWatch Container Insights

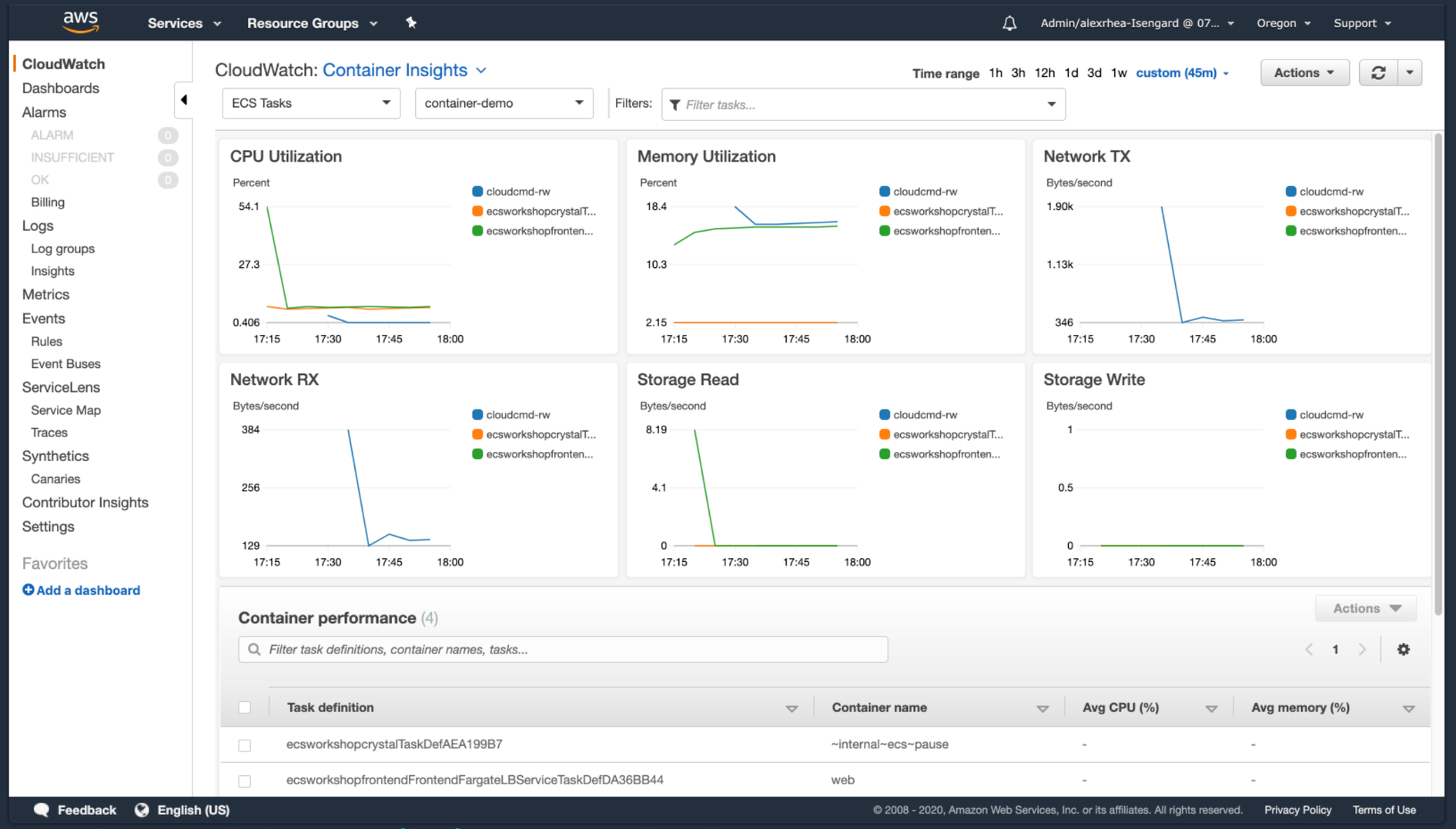
컨테이너화된 애플리케이션이나 마이크로서비스에 대한 모니터링, 트러블슈팅 및 알람을 위한 **완전 관리형 관측 서비스**

- 안정적이고 안전한 지표 및 로그 수집
- 자동화된 요약과 분석
- 지표, 로그, 트레이스 전반에 대한 가시성
- 사전 생성된 대시보드

https://ecs-cats-dogs.workshop.aws/ko/monitoring.html

aws

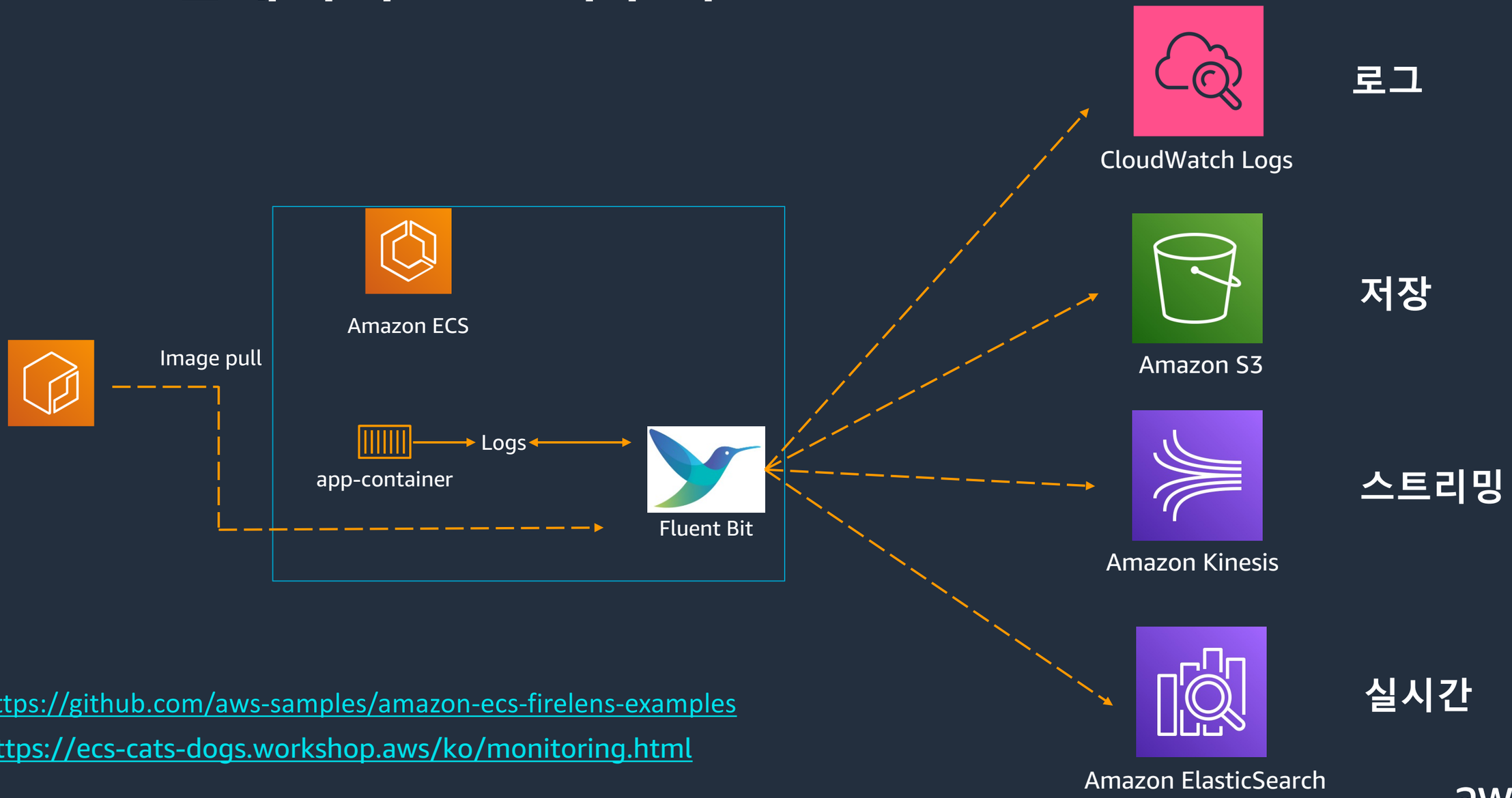
Amazon ECS CloudWatch Container Insights



https://ecs-cats-dogs.workshop.aws/ko/monitoring.html

aws

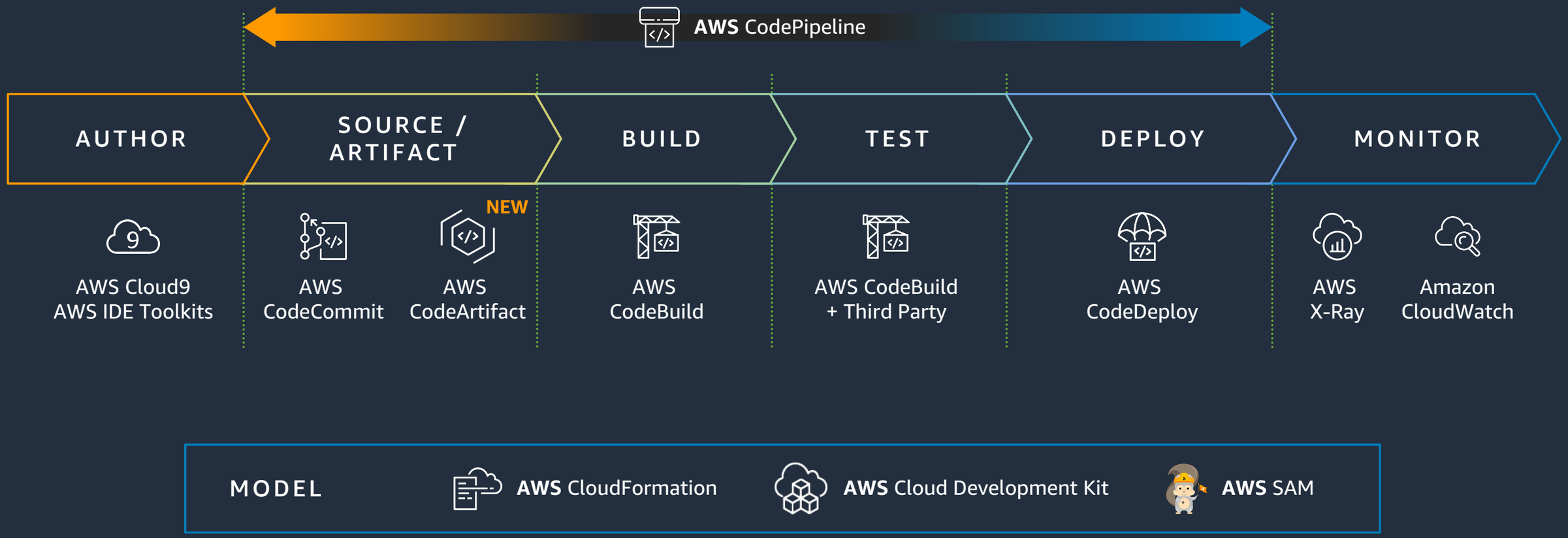
FireLens: 컨테이너 로그 라우터



https://github.com/aws-samples/amazon-ecs-firelens-examples

https://ecs-cats-dogs.workshop.aws/ko/monitoring.html

AWS Developer Tool



https://ecs-cats-dogs.workshop.aws/ko/cicd.html

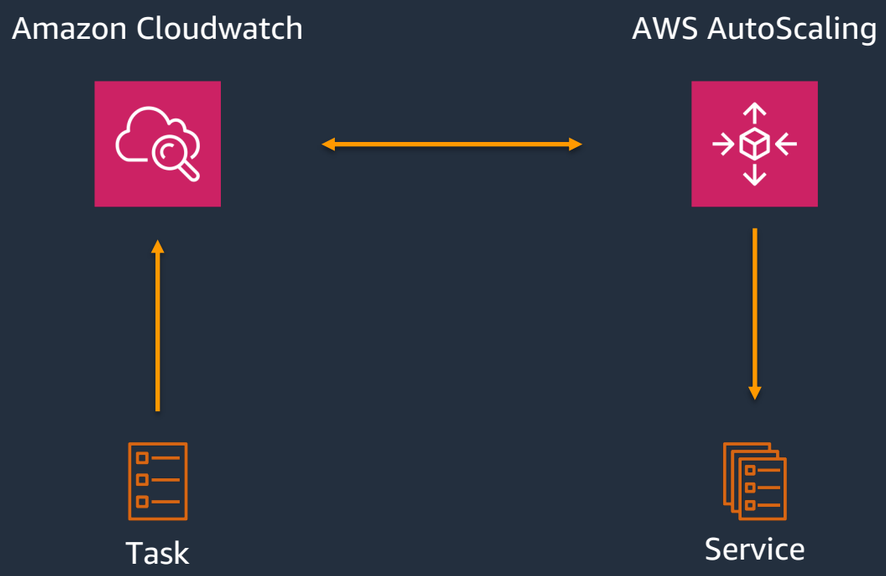
ECS Auto Scaling and Capacity Provider

https://ecs-cats-dogs.workshop.aws/ko/autoscale.html

aws

ECS Service Auto Scaling

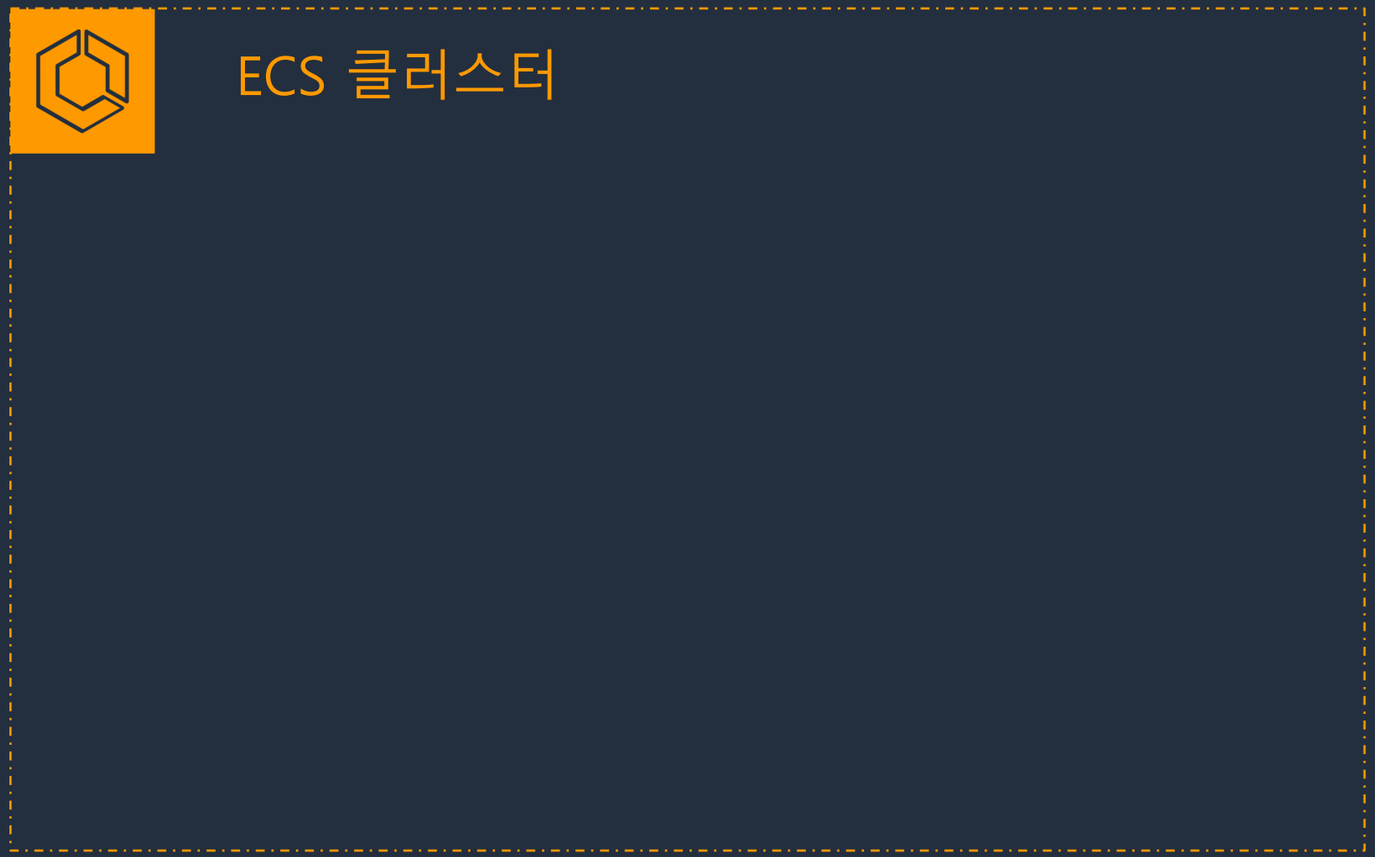
서비스 내의 작업 수를 자동으로 늘리거나 줄이는 역할을 함
ECS는 CPU 및 메모리 통계를 CloudWatch에 게시하고, Target Tracking, Step Scaling, 그리고 Scheduled Scaling을 지원함



https://ecs-cats-dogs.workshop.aws/ko/autoscale/service.html

aws

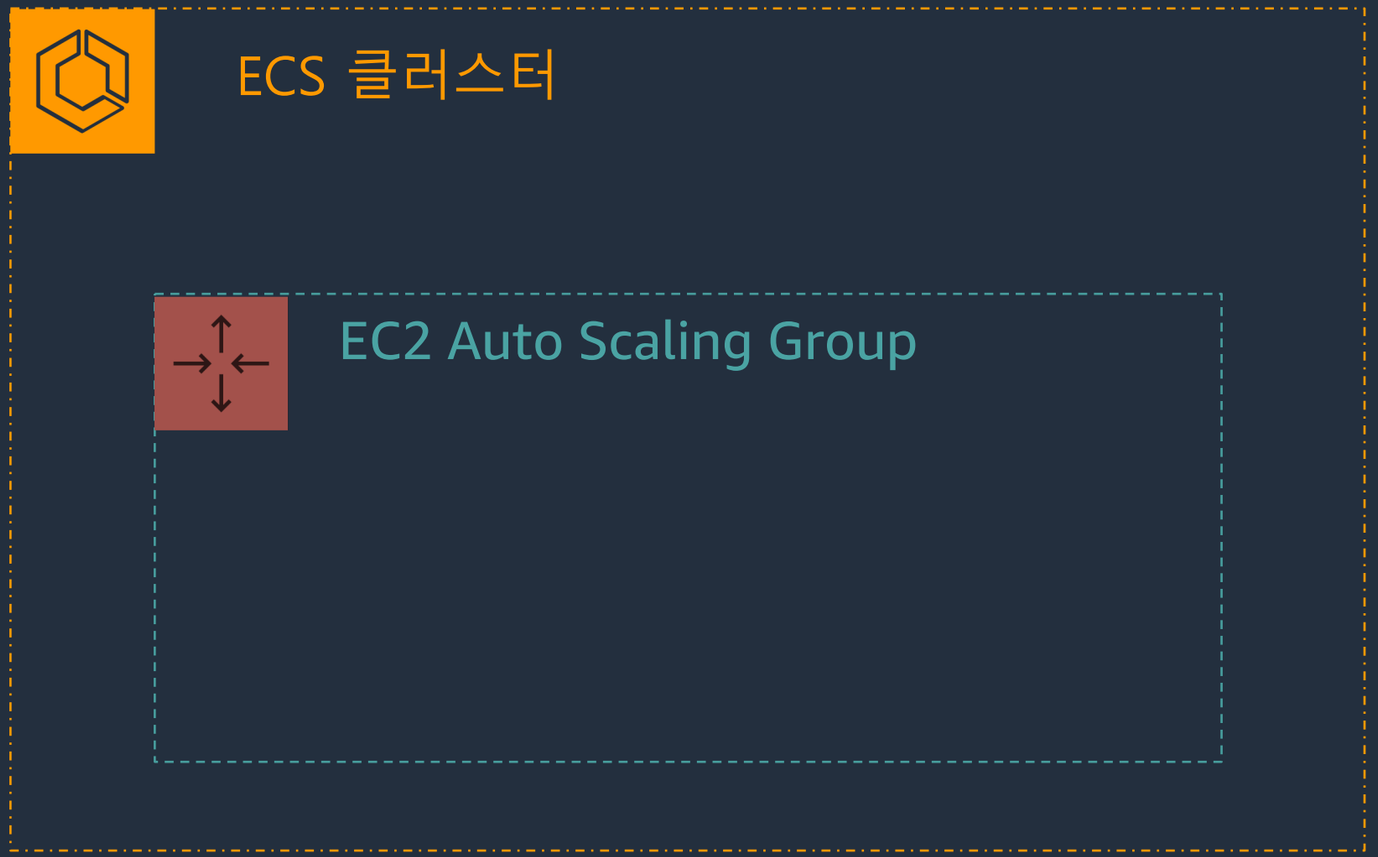
ECS 클러스터와 Amazon EC2 Auto Scaling 그룹



클러스터 생성

aws

ECS 클러스터와 Amazon EC2 Auto Scaling 그룹

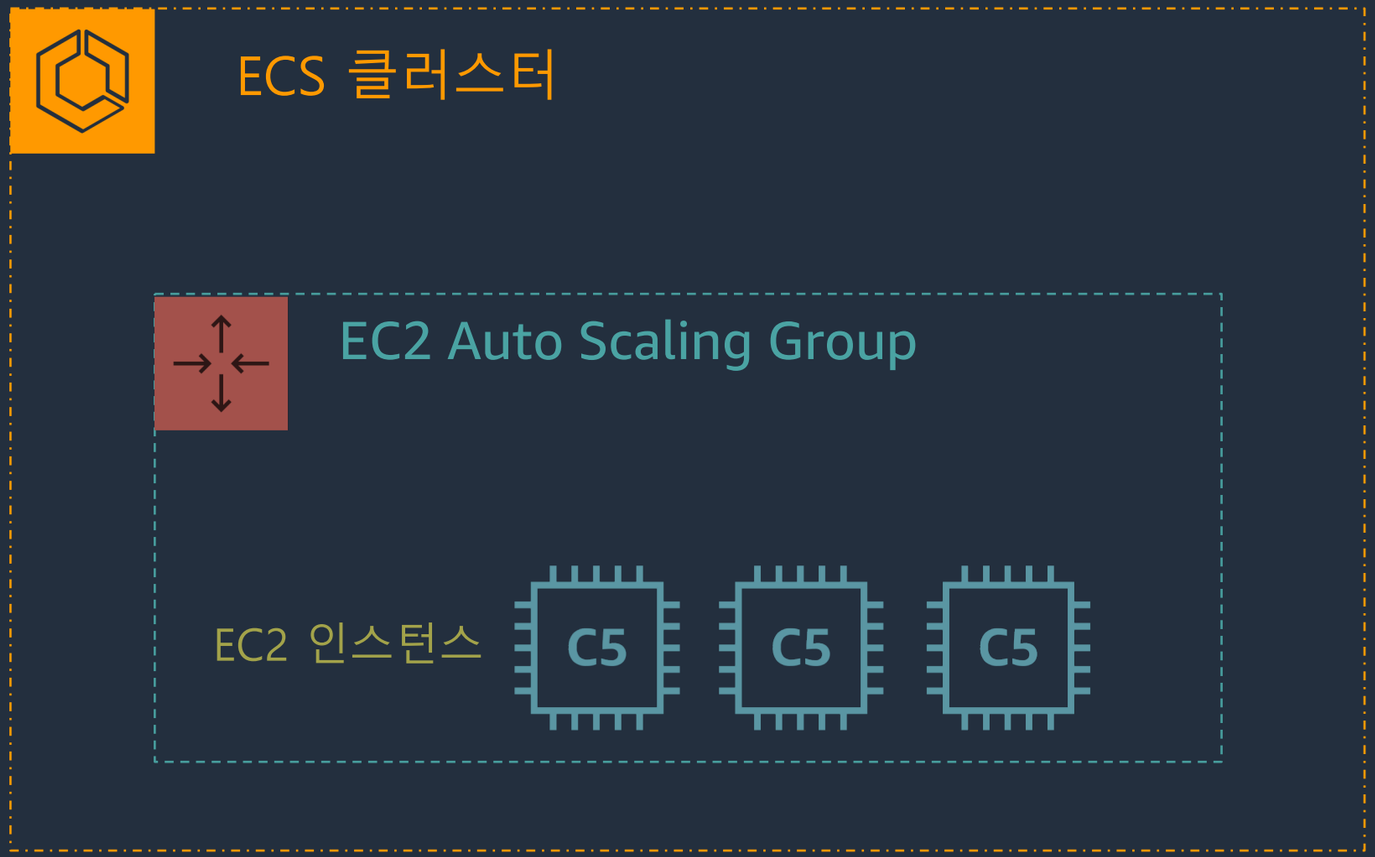


클러스터 생성

ASG 생성

aws

ECS 클러스터와 Amazon EC2 Auto Scaling 그룹



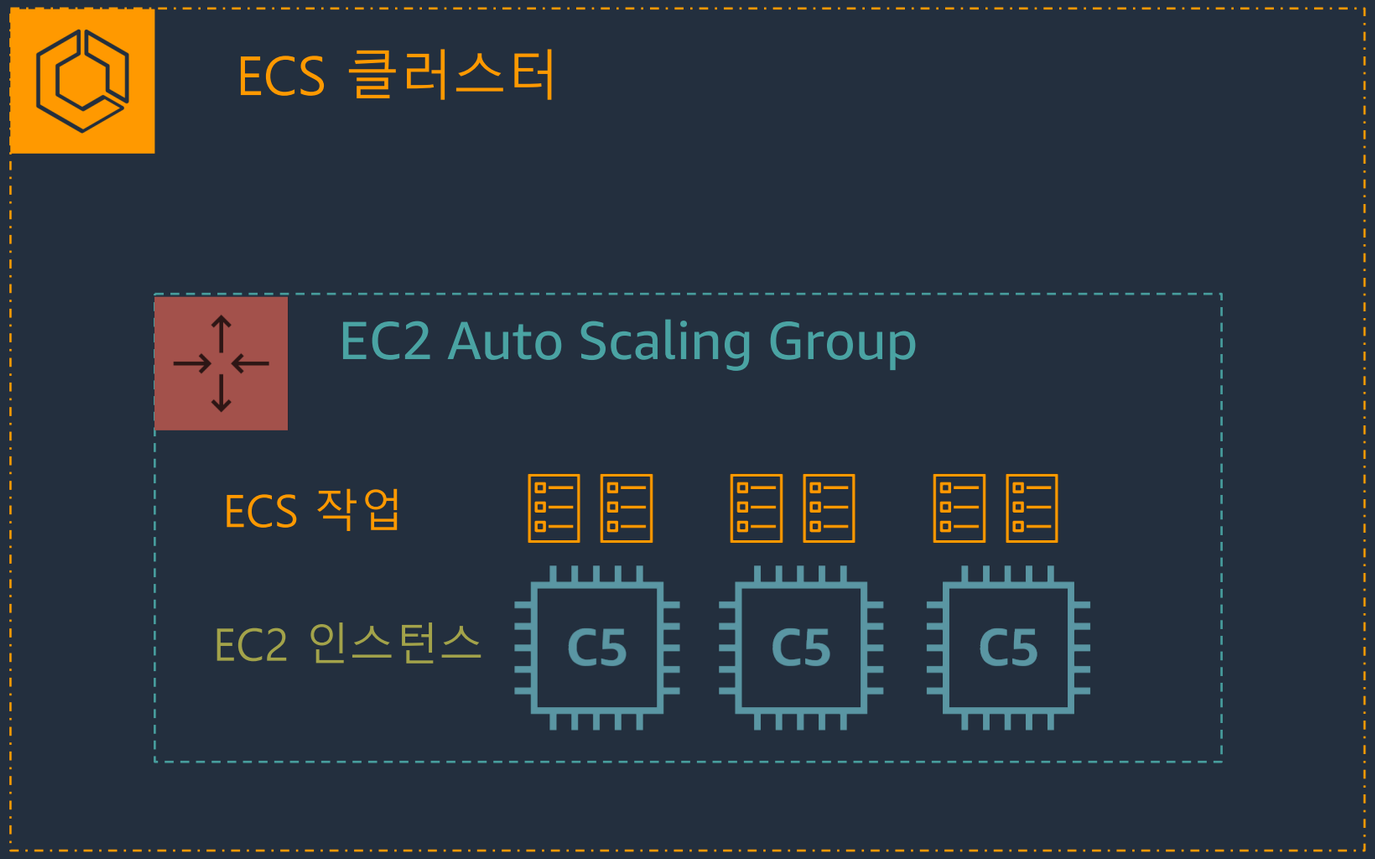
클러스터 생성

ASG 생성

인스턴스 시작

aws

ECS 클러스터와 Amazon EC2 Auto Scaling 그룹



클러스터 생성

ASG 생성

인스턴스 시작

작업 수행

aws

애플리케이션 우선

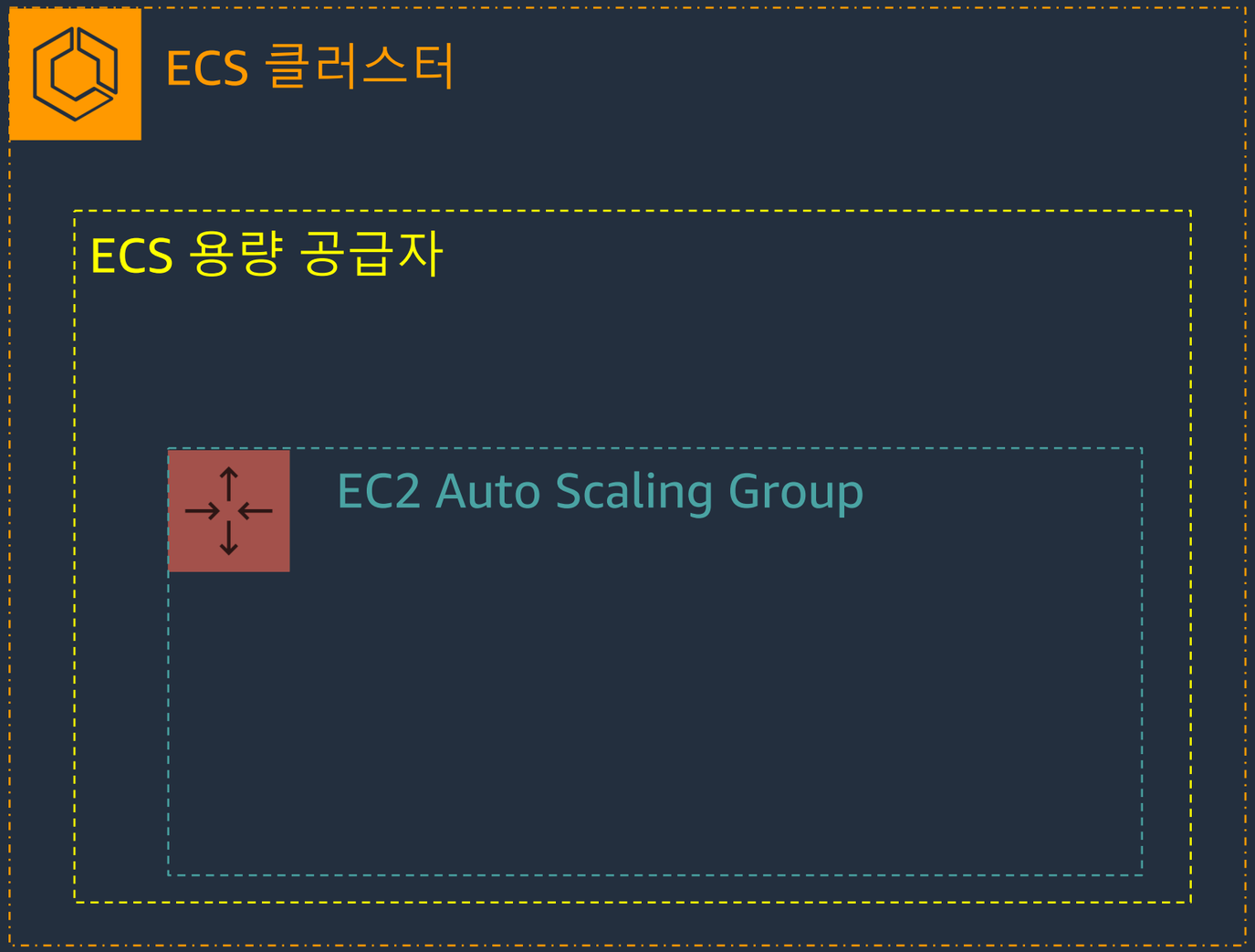
애플리케이션 우선 주의

애플리케이션은 각자의 요구사항이 있음

인프라스트럭처는 애플리케이션의 요구조건에 응답해야 함

aws

Amazon ECS 용량 공급자: 연결고리



클러스터 생성 (1회)

ASG 생성 (1회)

용량 공급자 생성(1회)

Amazon ECS 용량 공급자: 연결고리

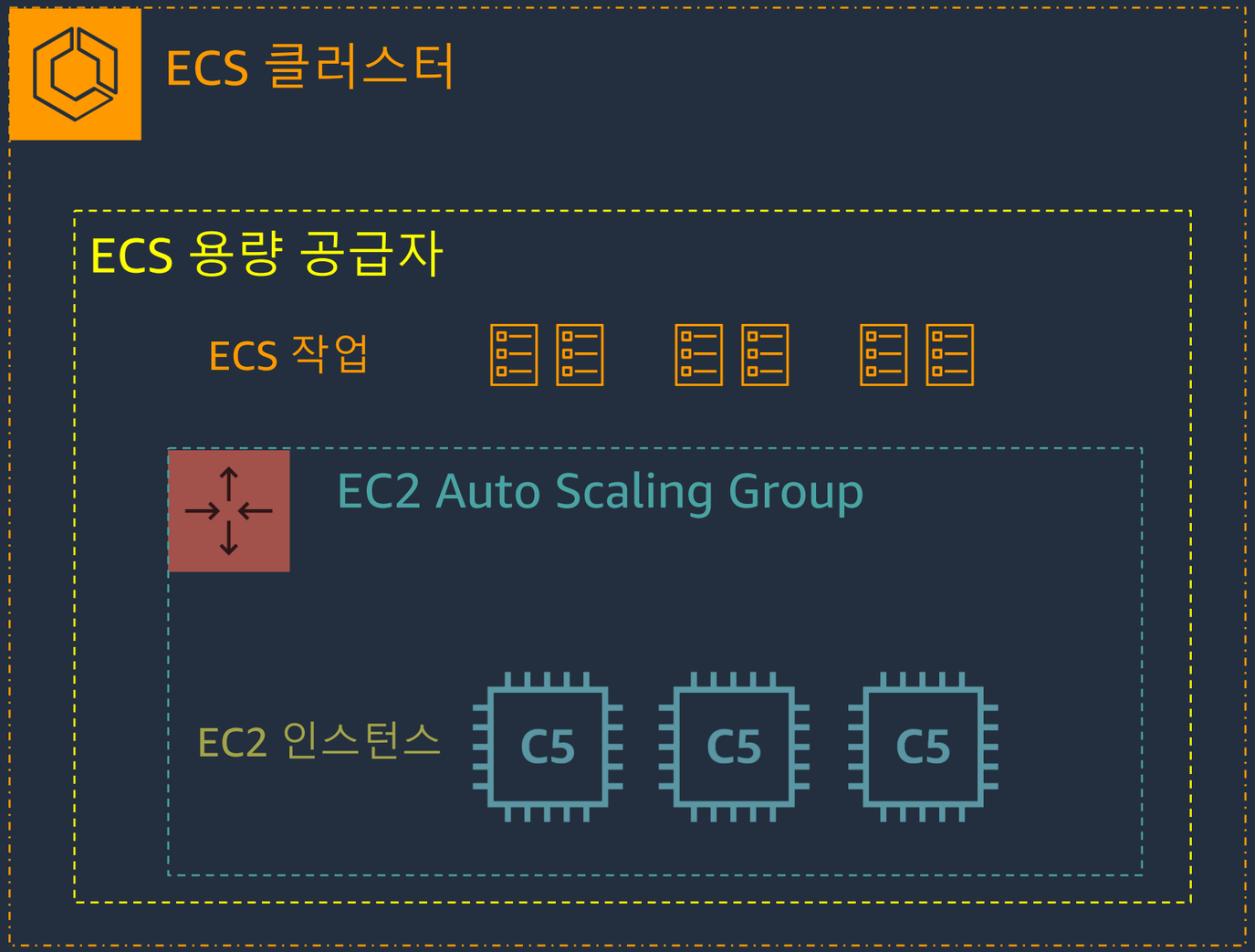
클러스터 생성 (1회)

ASG 생성 (1회)

용량 공급자 생성(1회)

작업 실행

Amazon ECS 용량 공급자: 연결고리



클러스터 생성 (1회)

ASG 생성 (1회)

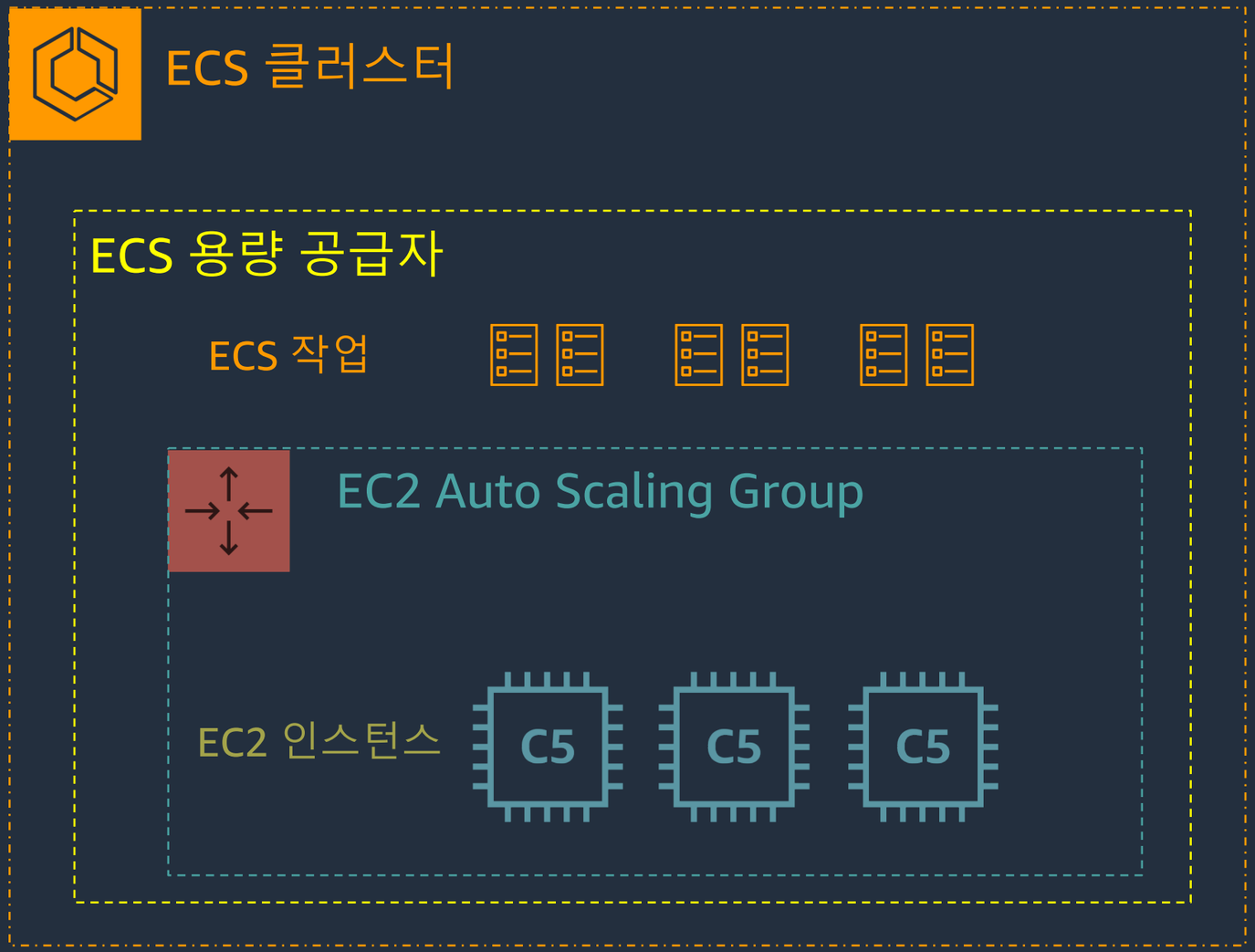
용량 공급자 생성(1회)

작업 실행

인스턴스 시작

aws

Amazon ECS 용량 공급자: 연결고리



클러스터 생성 (1회)

ASG 생성 (1회)

용량 공급자 생성(1회)

작업 실행

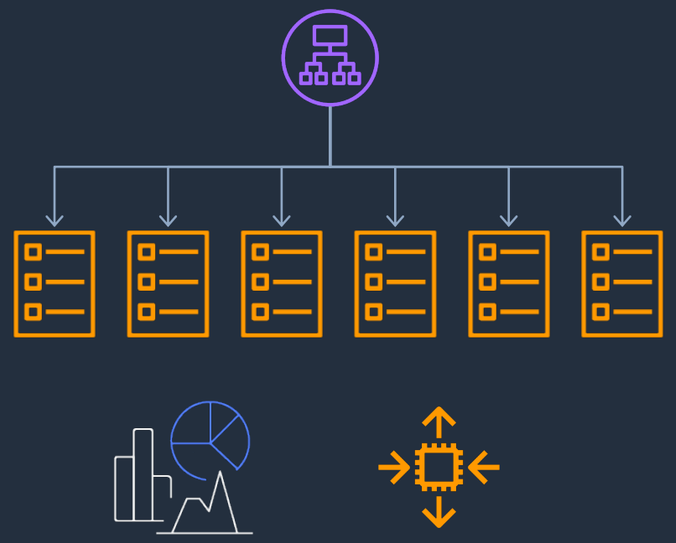
인스턴스 시작

작업 배치

aws

여러 용량 공급자로 분할: On-demand와 Spot

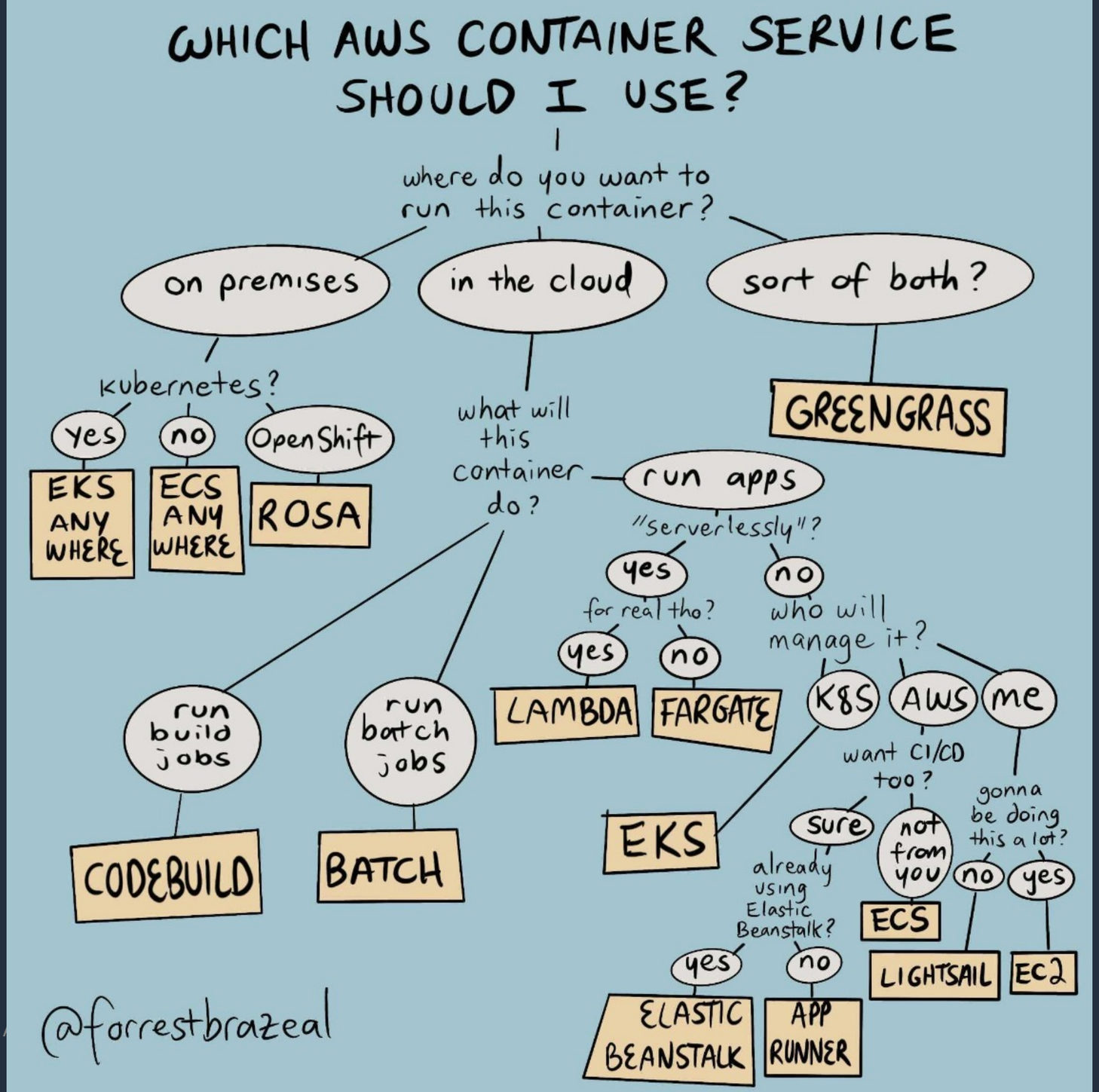
50% 여유 리소스 준비:
2/3 on-demand, 1/3 spot



+50% 여유
+5-10% 추가 비용

https://aws.amazon.com/ko/blogs/containers/deep-dive-on-amazon-ecs-cluster-auto-scaling/

aws



Start Containerization

aws

ECS Cats and Dogs 워크샵

https://ecs-cats-dogs.workshop.aws/ko/

컨테이너화를 쉽게 도와줄 도구들

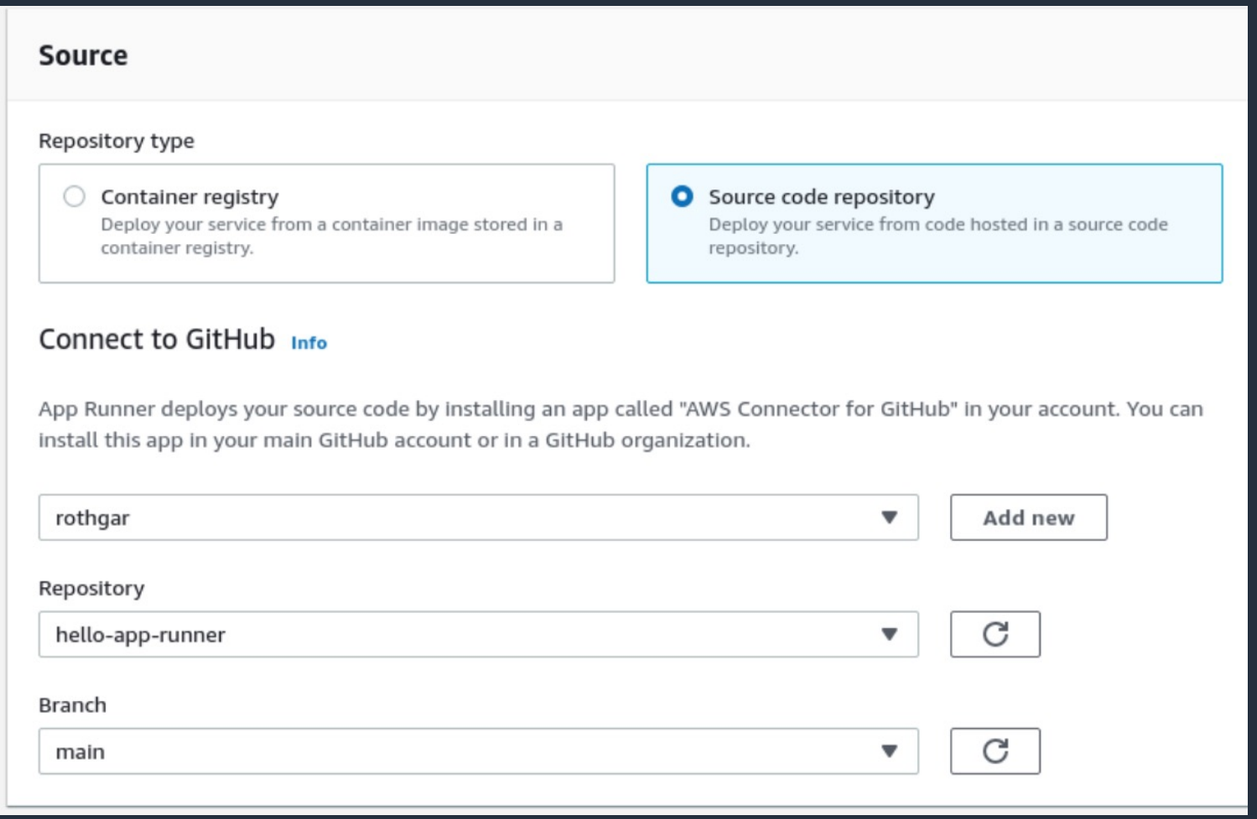
AWS App Runner

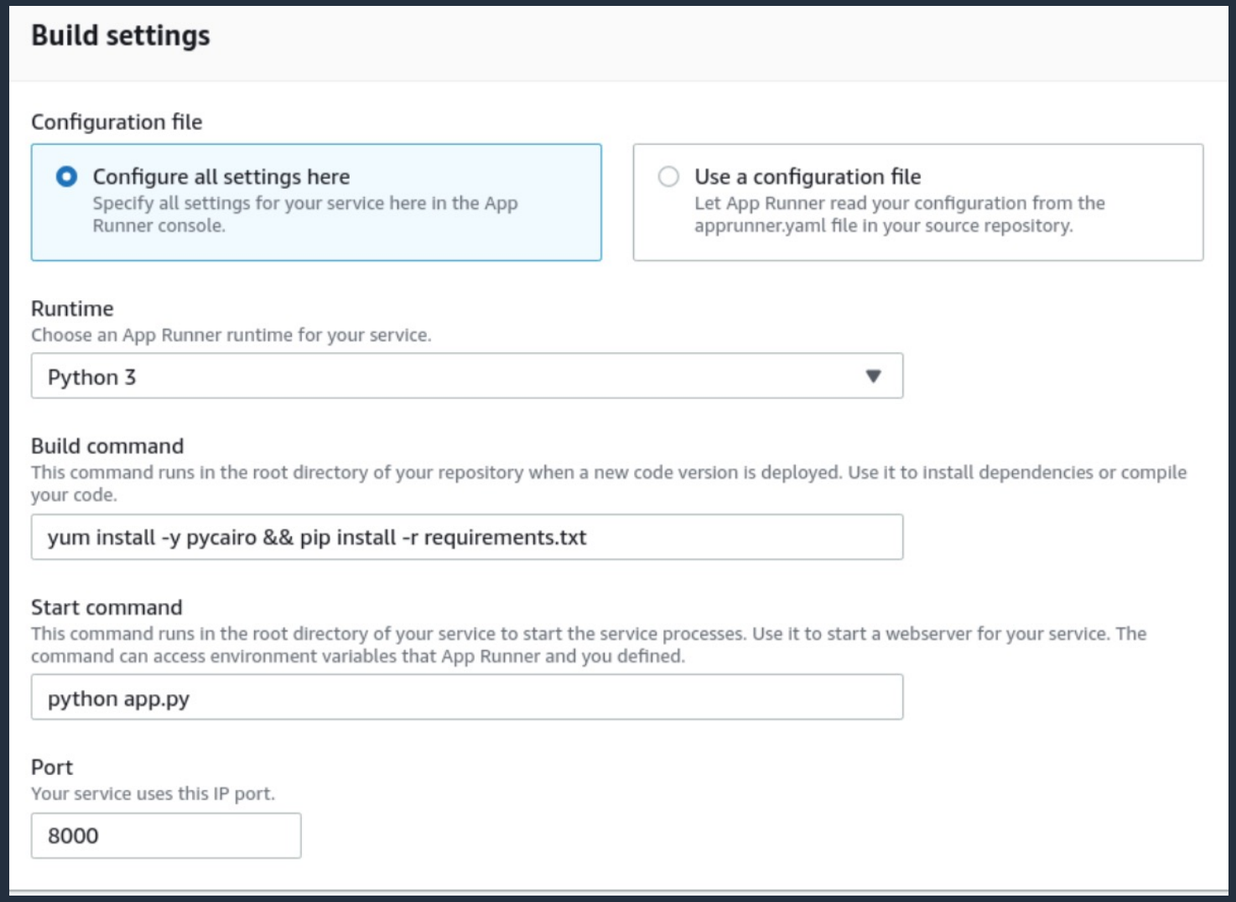
* 6월 기준 서울 리전 미지원

AWS Copliot CLI

aws

AWS App Runner





https://aws.amazon.com/ko/blogs/korea/app-runner-from-code-to-scalable-secure-web-apps/

aws

Copilot Resources

- Environment
- Application
- Service
- Task
- Pipeline
- Storage

```
Commands
  Getting Started 🌱
    init        Create a new ECS application.
    docs        Open the copilot docs.

  Develop ✨
    app         Commands for applications.
                Applications are a collection of services and environments.

    env         Commands for environments.
                Environments are deployment stages shared between services.

    svc         Commands for services.
                Services are long-running Amazon ECS services.

    job         Commands for jobs.
                Jobs are tasks that are triggered by events.

    task        Commands for tasks.
                One-off Amazon ECS tasks that terminate once their work is done.

  Release 🚀
    pipeline    Commands for pipelines.
                Continuous delivery pipelines to release services.

    deploy      Deploy a Copilot job or service.

  Addons 🧸
    storage     Commands for working with storage and databases.

  Settings ⚙️
    version     Print the version number.
    completion  Output shell completion code.

Flags
  -h, --help      help for copilot
  -v, --version   version for copilot

Examples
  Displays the help menu for the "init" command.
  `$ copilot init --help`
```

https://aws.amazon.com/ko/blogs/containers/introducing-aws-copilot/

aws

감사합니다 ☺

jimini@amazon.com