



AWS TECHCAMP 200

완전관리형 컨테이너 서비스 Amazon ECS로 애플리케이션 쉽게 구축하기

Jaemin Jung

Solutions Architect, AWS

AWS Builders Korea → AWS TechCamp로 새롭게 태어납니다.

AWS에서는 지난 3년 동안 고객분들의 적극적인 피드백을 반영하여 보다 집중적이고 쉽게 라이브 실습 세션을 제공하고자 새로운 이름과 보다 간결하고 쉬운 콘텐츠로 돌아왔습니다. AWS TechCamp는 온라인과 오프라인 세션으로 구성되어 있습니다. 참가자들은 실시간 이론+실습 세션을 통하여 AWS의 다양한 서비스와 기능을 테마별, 인더스트리별로 경험할 수 있습니다.

온라인 세션

분기별 3일 과정의 세션

AWS TechCamp 온라인 세션의 경우 3월 (Modern App) 6월 (AI/ML), 9월 (Data), 11월 (Every App) 총 4회 제공합니다. 세션은 3일 동안 진행되며 클라우드 클라우드 서비스가 생소한 분들을 위한 기초 과정 (레벨 100), 클라우드 기본 서비스 교육이 필요하신 분들을 위한 기본 과정 (레벨 100-200)으로 이루어져 있습니다. 자세한 내용은 아래 최신 세션 목록을 참고해 주세요.

오프라인 세션

특정 인더스트리 세션

TechCamp 오프라인 세션은 특정 인더스트리에 대한 주제로 초청 고객 대상 집중 세션으로 구성된 기본/심화과정 (레벨 200-300)이 진행됩니다.

관련 문의는 aws-korea-event@amazon.com으로 연락 주시길 바랍니다.

Agenda

- Amazon ECS
- AWS Fargate
- Amazon ECS 오브젝트
- Amazon ECS 네트워크
- AWS Developer Tool

Amazon ECS



Amazon ECS

AWS의 컨테이너 서비스



ECS

강력한 단순성



EKS

개방적 유연성

Amazon ECS

강력한 단순성



ECS

AWS에서 제시하는 대규모
컨테이너 실행 방법

규모나 기능을 조정할 필요 없음

애플리케이션 구축, 배포 및
마이그레이션 시간 단축

Amazon ECS

고객이 AMAZON ECS를 사용하는 세 가지 대표적인 이유



시장 출시 시간 단축

낮은 운영 오버헤드
업그레이드가 필요 없고
배우기 쉬운
컨트롤 플레인



비용 절감

사용량에 따라
지불하는 Fargate
Scale을 사용하여
0에서 대량으로



보안 및 규정 준수

정의된 보안 모델 및 규정 준수
AWS 통합

Amazon ECS

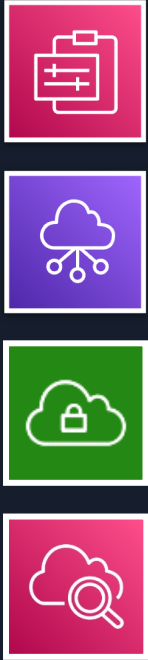
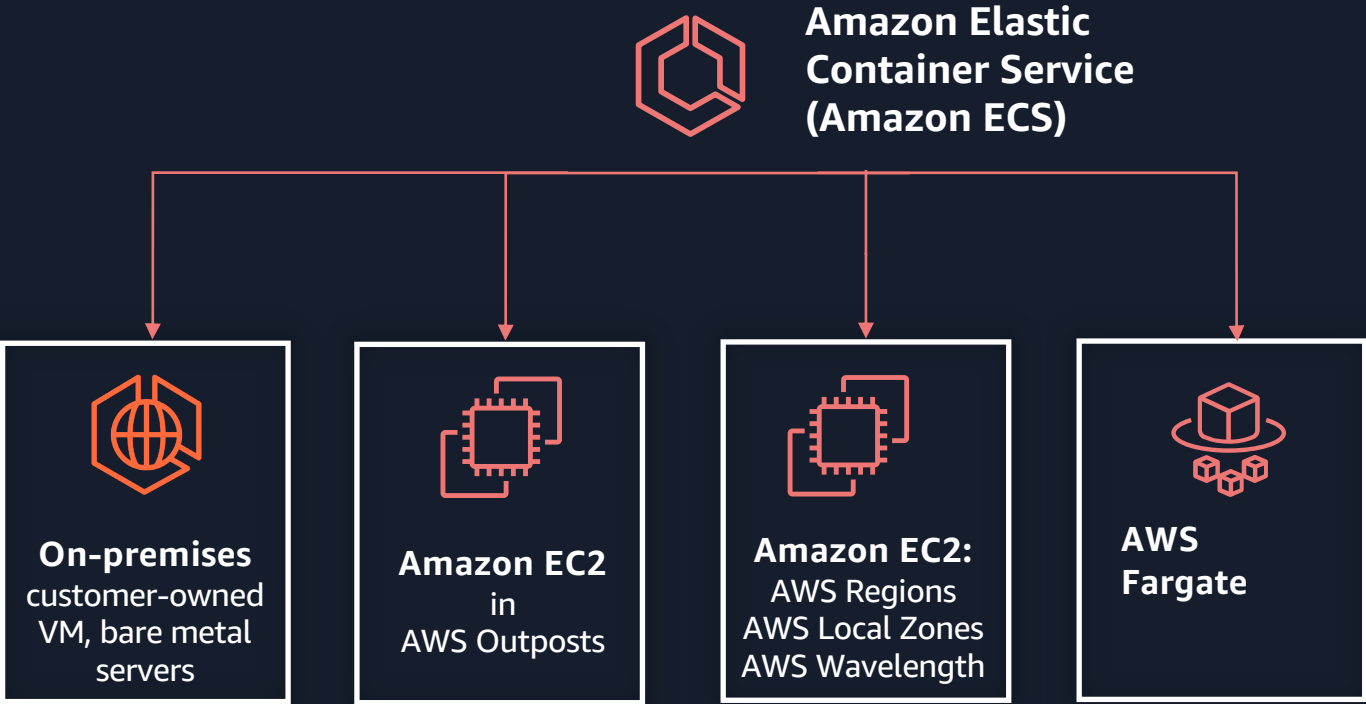
AWS 컨테이너 서비스 포트폴리오

컨트롤 플레인

배포, 스케줄링, 컨테이너화된
애플리케이션 스케일링&관리

데이터 플레인

컨테이너가 동작하는 환경



← On-premises Managed in the cloud →

Amazon ECS

AMAZON ECS의 정의

Amazon ECS는 서버리스 컨테이너 오케스트레이터이다.



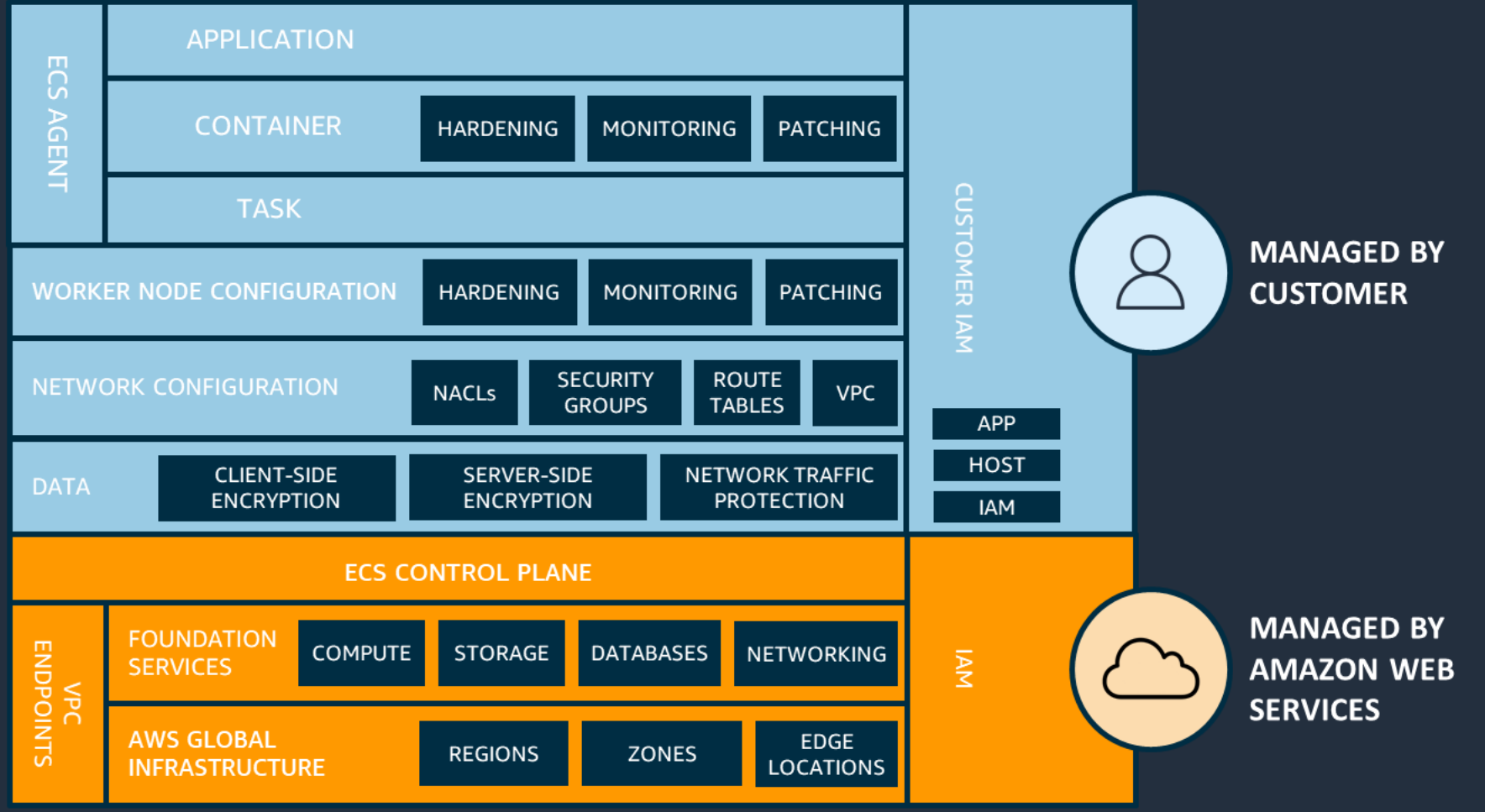
AWS에서 관리하며
클러스터 버전이나 데이터
스토어를 유지 관리하거나
확장할 필요가 없으며
Amazon EC2 또는 AWS
Fargate에서 사용할 수
있습니다.



AWS Fargate를 비롯한
다양한 “컴퓨팅”
환경에서 컨테이너를
오케스트레이션합니다.

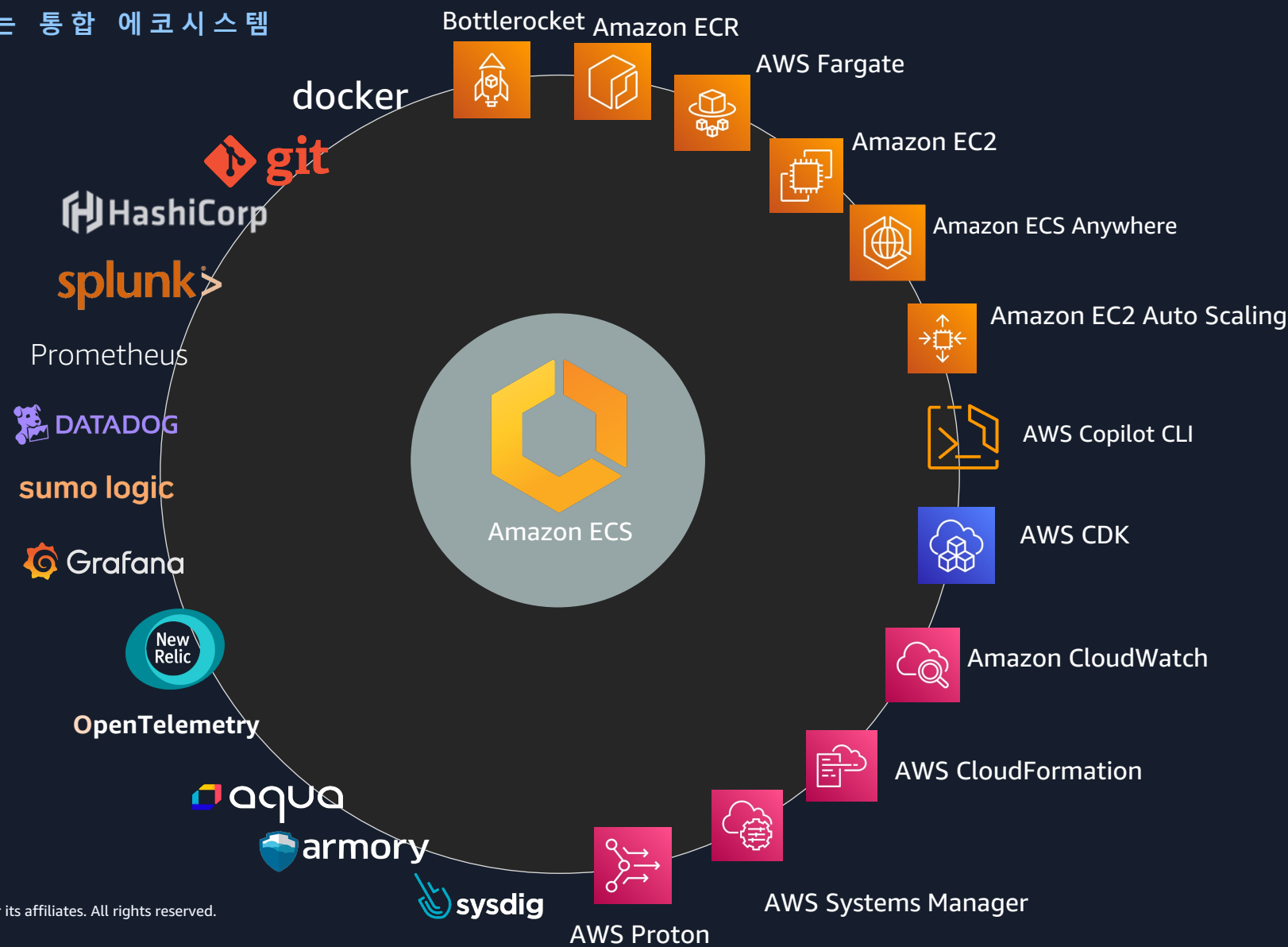
Amazon ECS

공동 책임 모델 - ECS/EC2



Amazon ECS

AMAZON ECS를 지원하는 통합 에코시스템

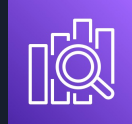


Amazon ECS

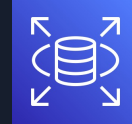
AMAZON ECS POWERS AMAZON



Amazon Redshift



Amazon OpenSearch Service



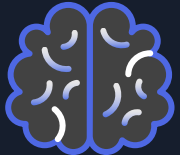
Amazon RDS



Amazon Transcribe



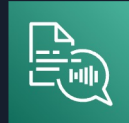
AWS Elemental
MediaLive



Amazon.com's
recommendation



Amazon Lex



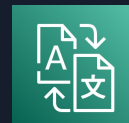
Amazon Polly



AWS Batch



Amazon SageMaker



Amazon Translate

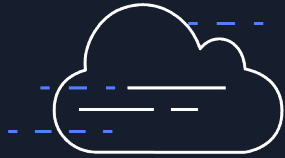
- Amazon ECS는 Amazon의 다양한 서비스를 위한 빌딩 블록을 형성합니다.
- 보안, 안정성, 가용성 및 확장성 테스트를 거쳤습니다.

AWS Fargate

AWS Fargate

AWS FARGATE의 정의

Fargate는 서버리스 컨테이너를 위한 컴퓨터 엔진이다



AWS에서 관리하므로
AMI를 유지 관리할
필요도 없고 EC2
인스턴스를 프로비저닝,
확장 또는 관리할 필요도
없습니다. 필요한 만큼만
비용을 지불하면 됩니다.



AWS Fargate는
컨테이너가 실행되는
곳입니다. 고객은 직접
상호 작용하지
않습니다.

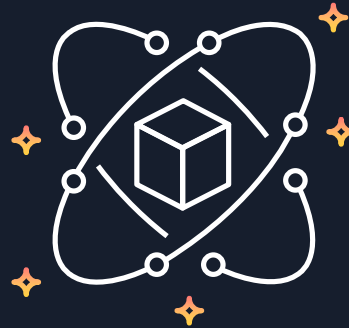
AWS Fargate

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

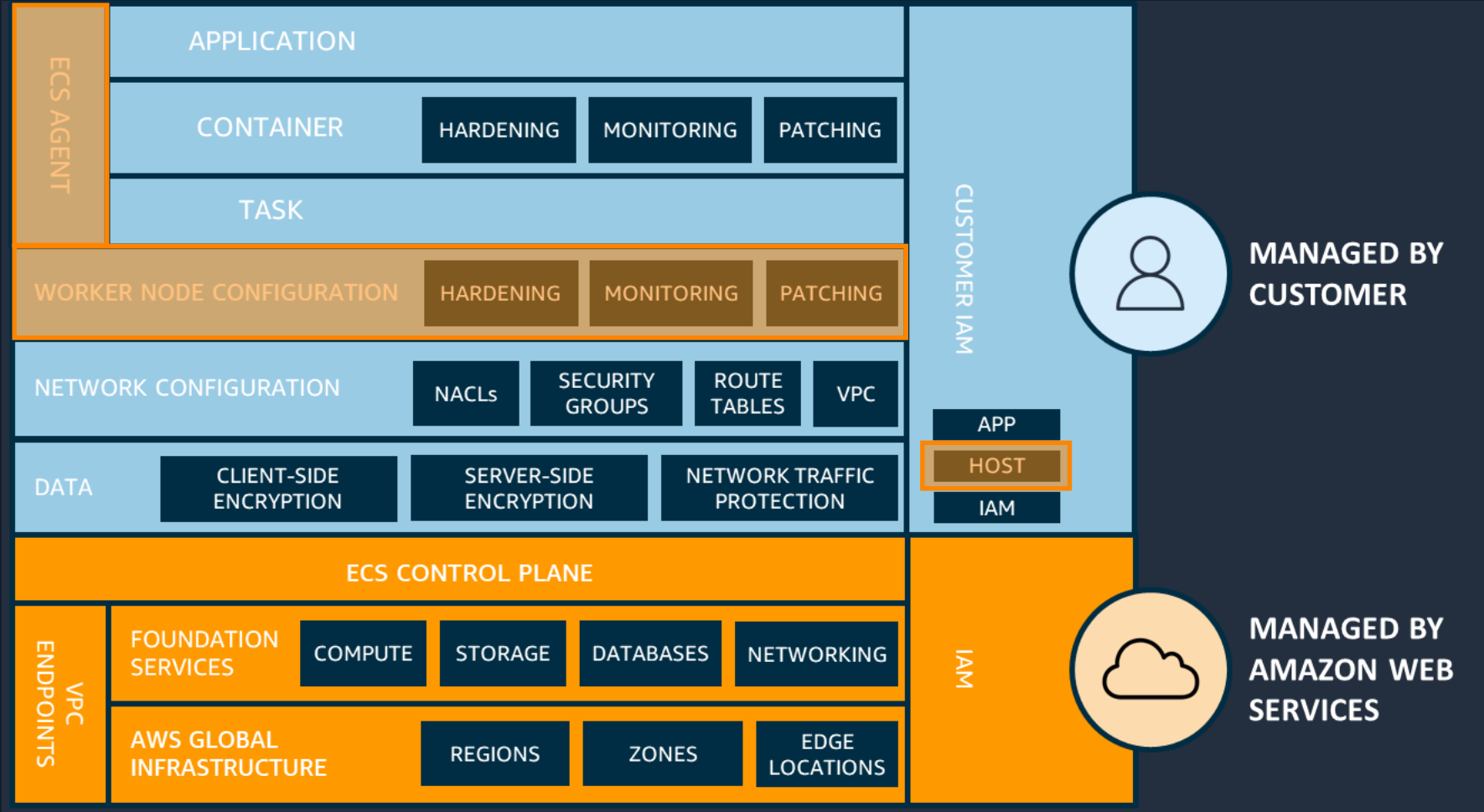


필요한 만큼만 & 쉬운 연동

컨테이너 실행에 필요한 자원만큼만
비용 지불
기존 AWS 네트워크, 보안과
네이티브하게 통합하여 사용

AWS Fargate

공동 책임 모델 - ECS/FARGATE



AWS Fargate

AWS FARGATE 작업 CPU와 메모리

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
8192 (8 vCPU)	16 GB to 60 GB
16384 (16 vCPU)	32 GB to 120 GB

- 0.5 vCPU부터는 메모리 할당이 1GB 단위로 가능
- 8 vCPU는 메모리 할당이 4 GB 단위로 가능
- 16 vCPU는 메모리 할당이 8 GB 단위로 가능



```
{
  "family": "sample-fargate",
  "networkMode": "awsvpc",
  "containerDefinitions": [
    {
      "name": "fargate-app",
      "image": "httpd:2.4",
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80,
          "protocol": "tcp"
        }
      ],
      "essential": true,
      "entryPoint": [
        "sh",
        "-c"
      ],
      "command": [
        "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top: 40px; background-color: #333;}</style> </head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon ECS.</p> </div></body></html>' > /usr/local/apache2/htdocs/index.html && httpd-foreground\""
      ]
    }
  ],
  "requiresCompatibilities": [
    "FARGATE"
  ],
  "cpu": "256",
  "memory": "512"
}
```

AWS Fargate

AWS FARGATE를 도입하기 위해 고려해야 될 사항

- 호스트 OS 단에 구축해야하는 compliance 및 소프트웨어 요소가 있다면 사용 불가능
- GPU 지원 X
- 작업 정의 시, 네트워크 모드 중, awsvpc 모드만 지원
- 작업(Task) ENI에 고정 IP 주소 사용 불가능 (아래 nlb 활용)

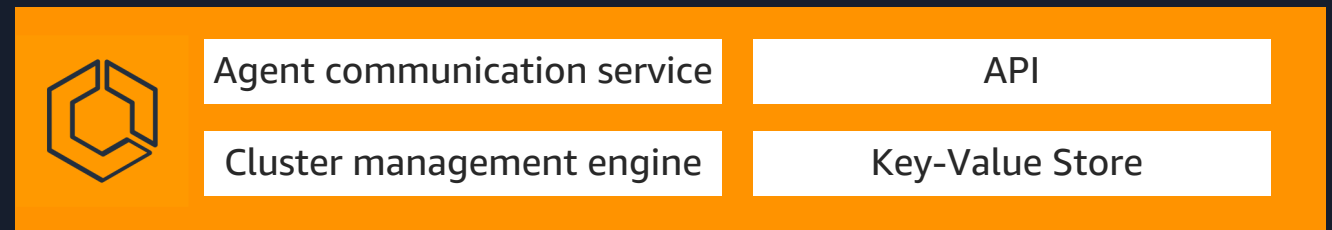
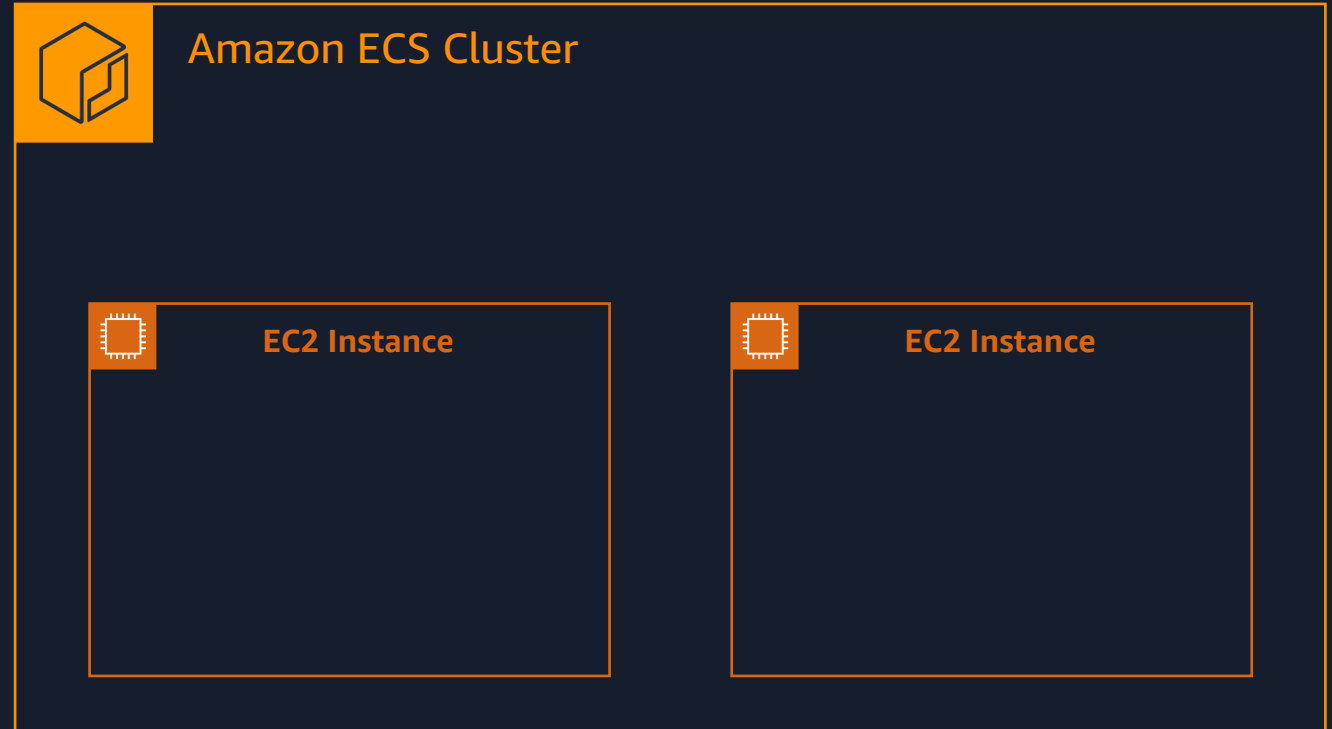
https://aws.amazon.com/ko/premiumsupport/knowledge-center/ecs-fargate-static-elastic-ip-address/?nc1=h_ls

Amazon ECS 오브젝트

Amazon ECS 오브젝트

AMAZON ECS 클러스터

- 작업이 실행되는 논리적인 그룹
- 컨트롤 플레인 / 데이터 플레인으로 나뉘어 구성



Amazon ECS 오브젝트

AMAZON ECS 작업

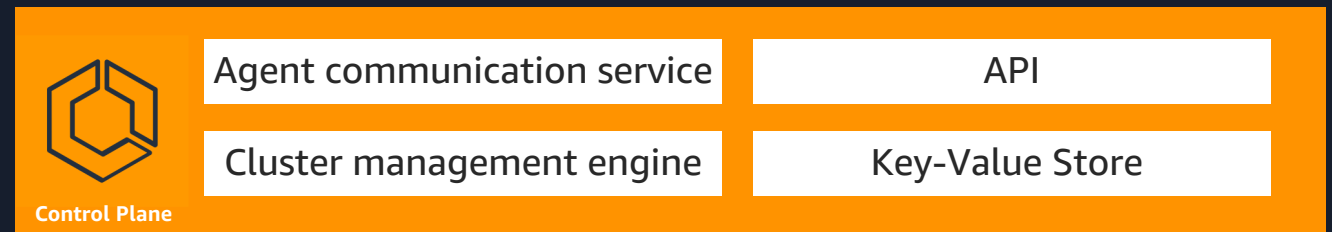
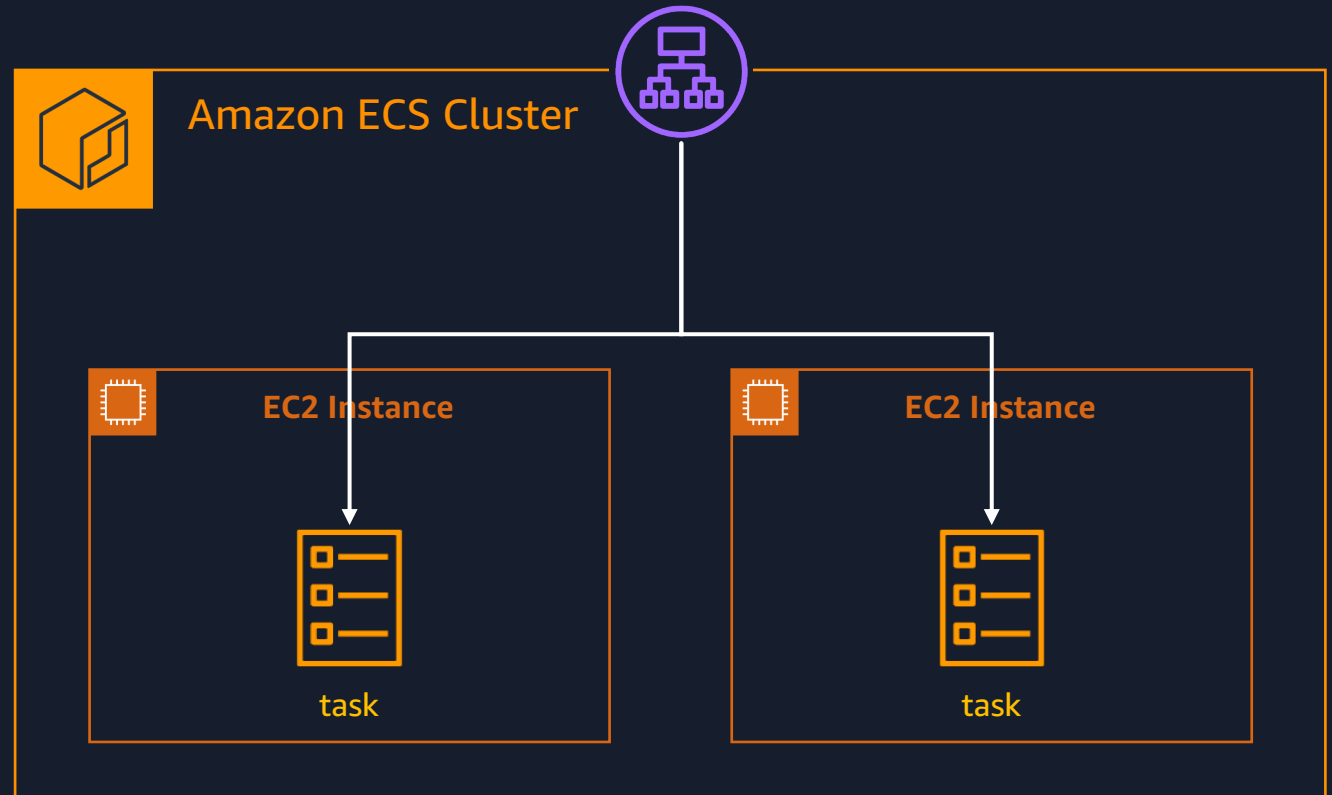
- ECS 클러스터에서 **최소 실행 단위**
 - 네트워킹, 스토리지, 파라미터, IAM 역할, 컴퓨팅 리소스 등의 구성 값 설정 가능
- **작업 정의(Task Definition)** 내용을 기반으로 작업 배포
 - 배포 타입 설정: Fargate, EC2, External
 - 컨테이너 이미지 매핑을 통한 정의 작업
 - 작업 역할을 부여해 API 요청을 받을 때 권한에 따라 동작
 - 작업 크기 설정
 - AppMesh, FireLens 등과의 통합 설정
- 하나의 작업 당 최대 10개 컨테이너 가능



Amazon ECS 오브젝트

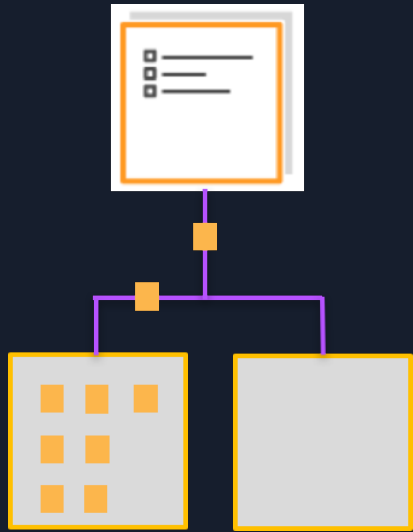
AMAZON ECS 서비스

- ECS 위에서 작업을 실행하는 방법
- 주요 설정 내용
 - 서비스 유형(replica, daemon)
 - 작업 배치 전략
 - 배포 유형
- AWS ELB를 통한 효율적 부하 분산
- Service Auto Scaling을 통한 탄력적 워크로드 구성

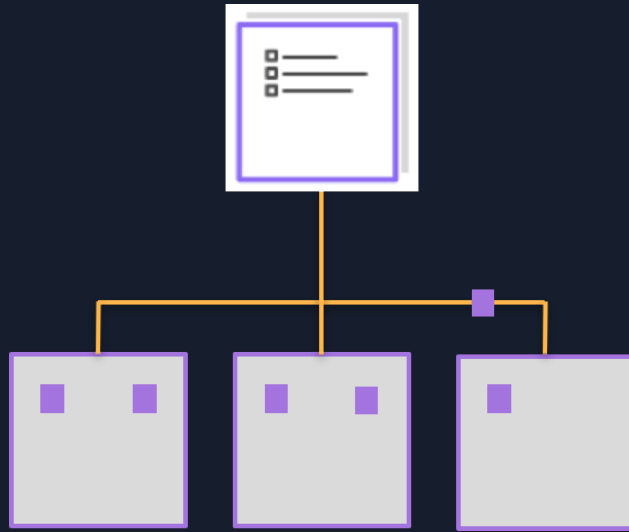


Amazon ECS 오브젝트

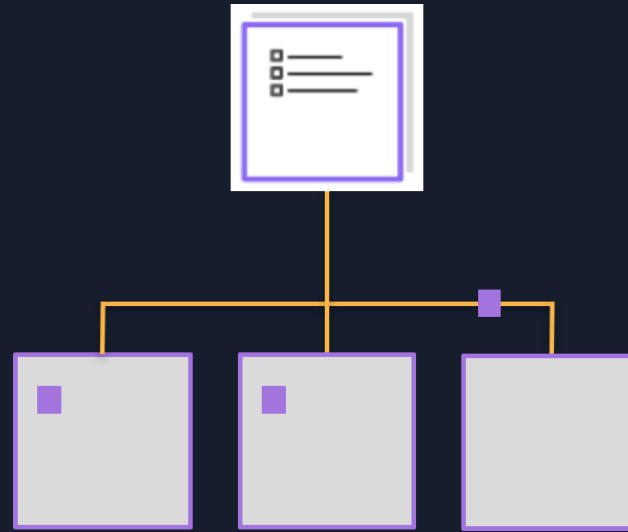
다양한 작업 배치 전략



Binpacking



Spread

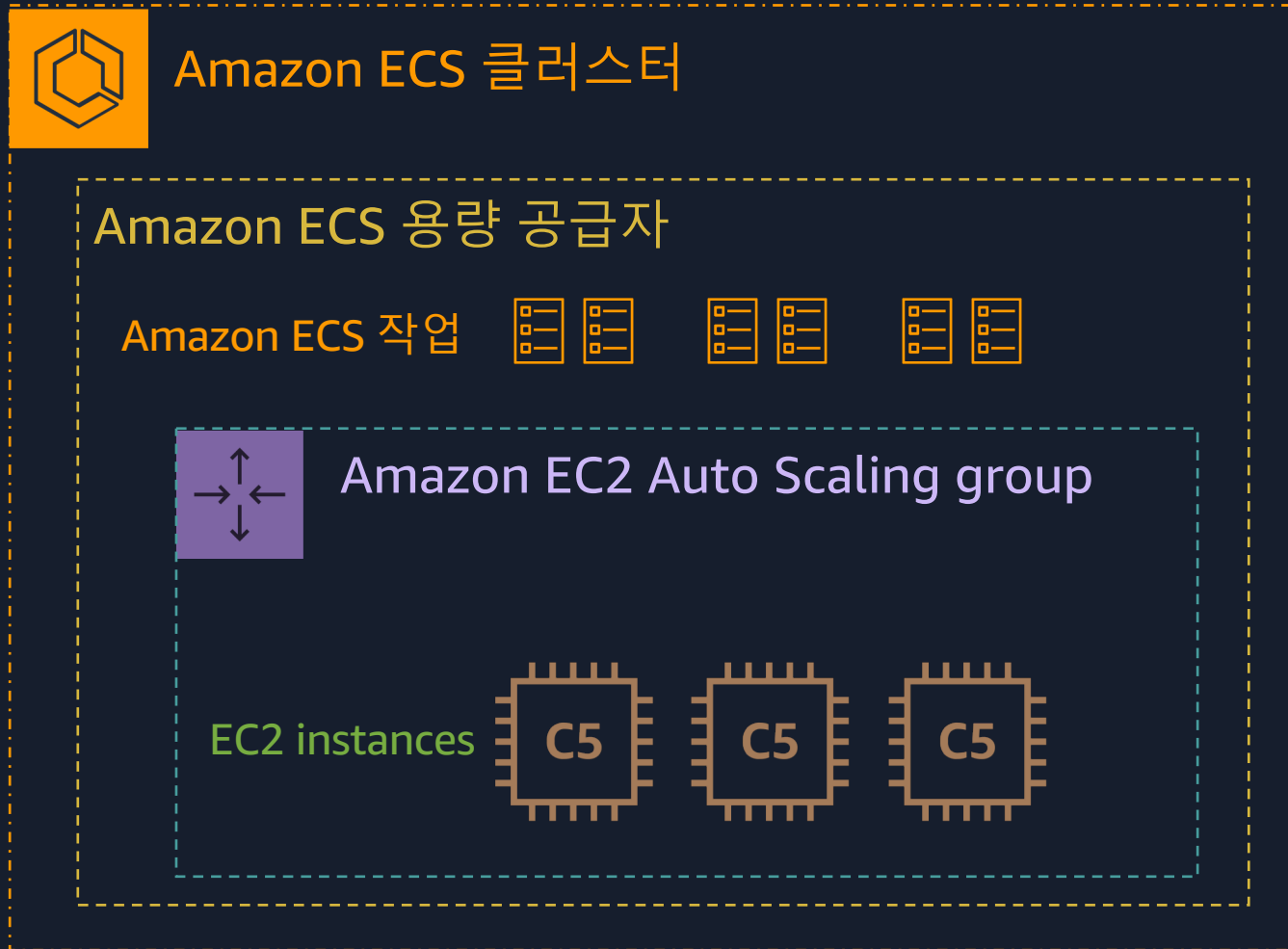


One Task per Host

...

Amazon ECS 오브젝트

AMAZON ECS 용량 공급자

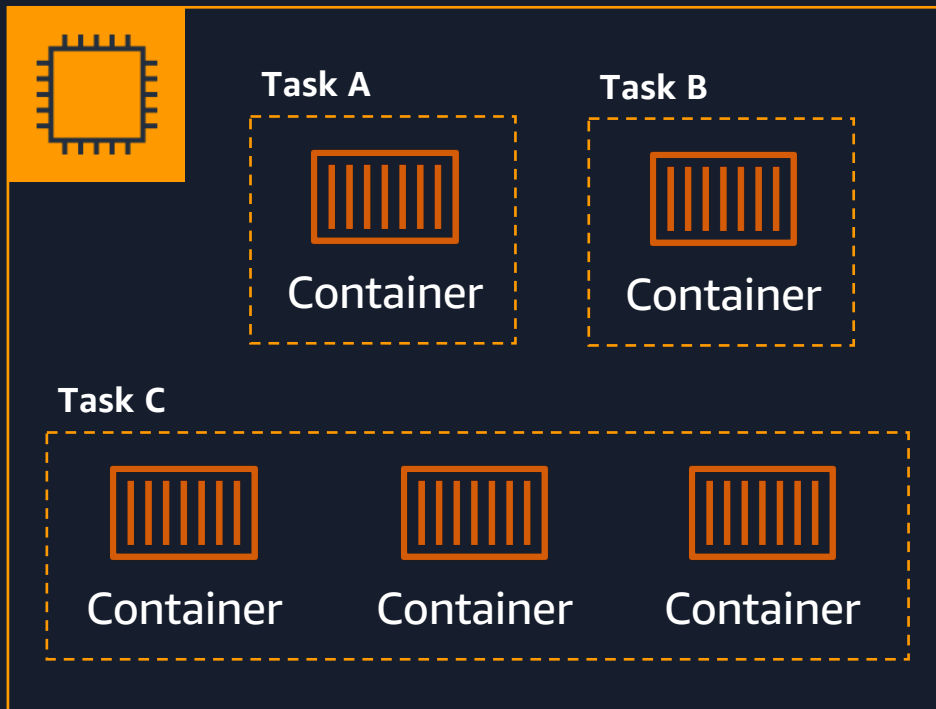


Amazon ECS 오브젝트

AMAZON ECS 용량 공급자 - EC2

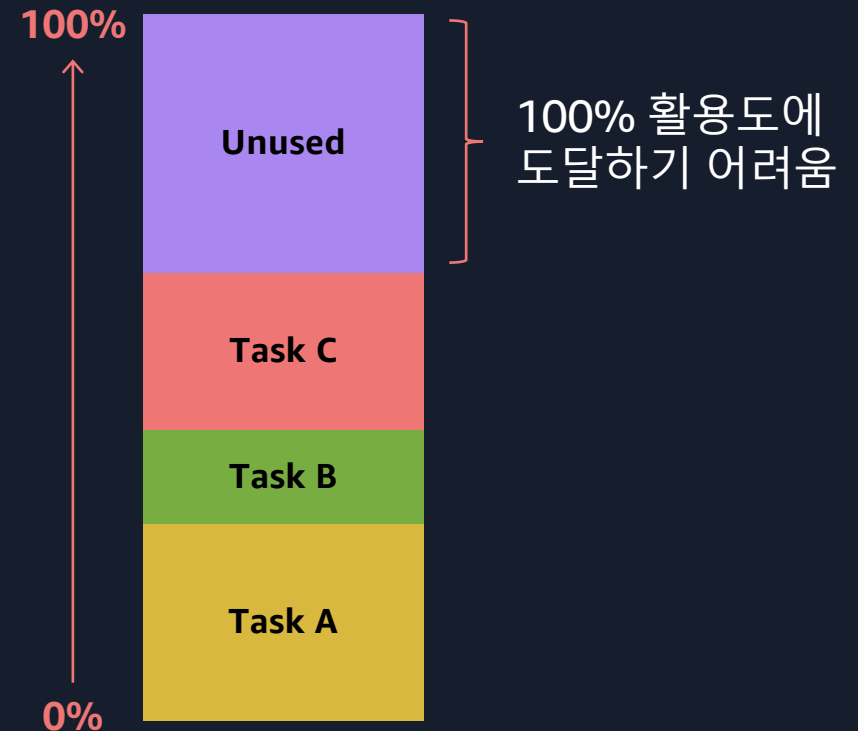
EC2 노드

컨테이너 수와 리소스 용량의 균형을 맞추기가 어려움



리소스 활용

여러 클러스터 노드에서 대량의 미사용 EC2 리소스에 대해 비용을 지불하는 경우가 많음



Amazon ECS 오브젝트

AMAZON ECS 용량 공급자 - FARGATE

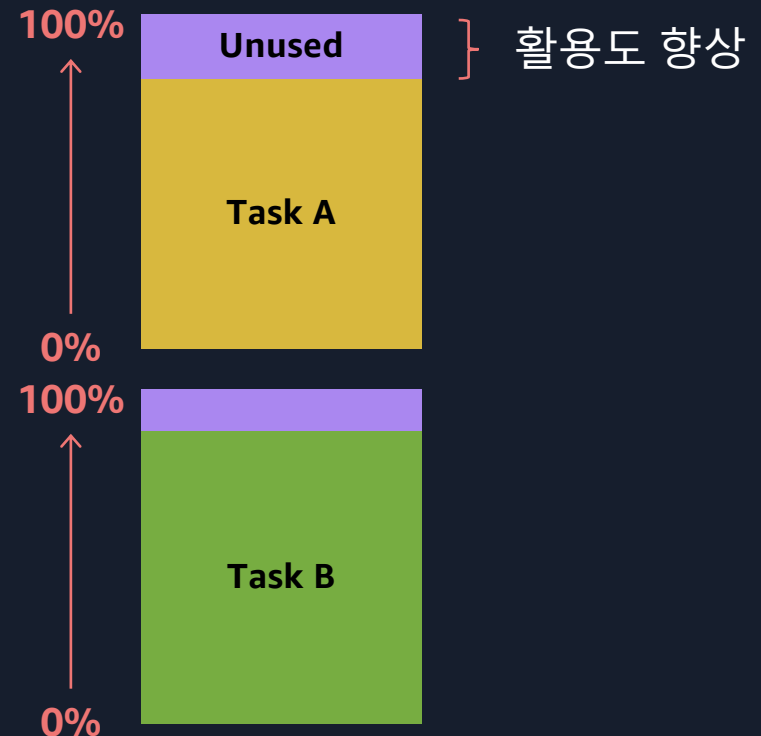
Fargate 작업

필요한 메모리 및 vCPU 리소스를 기반으로 각 작업의 크기를 조정합니다.



리소스 활용

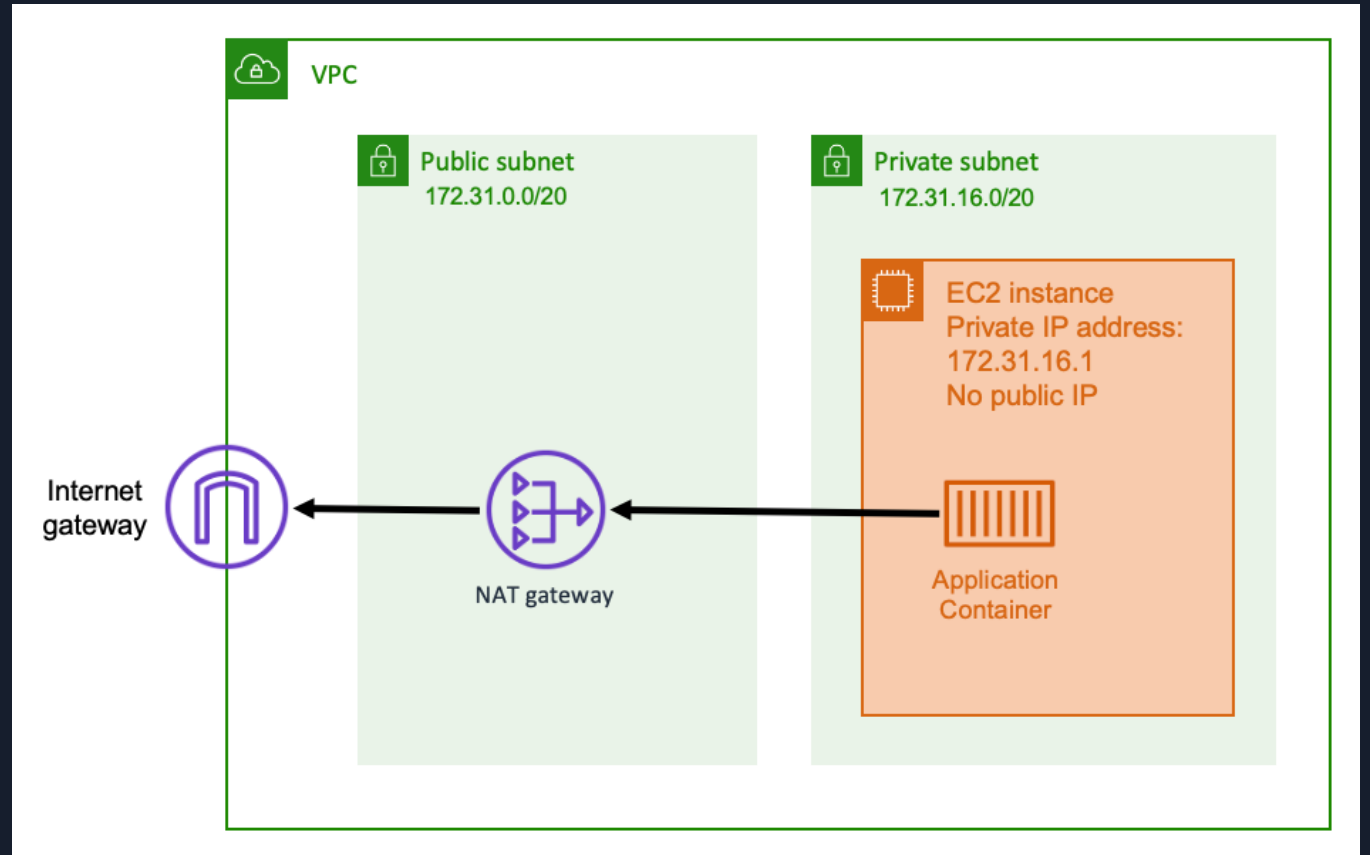
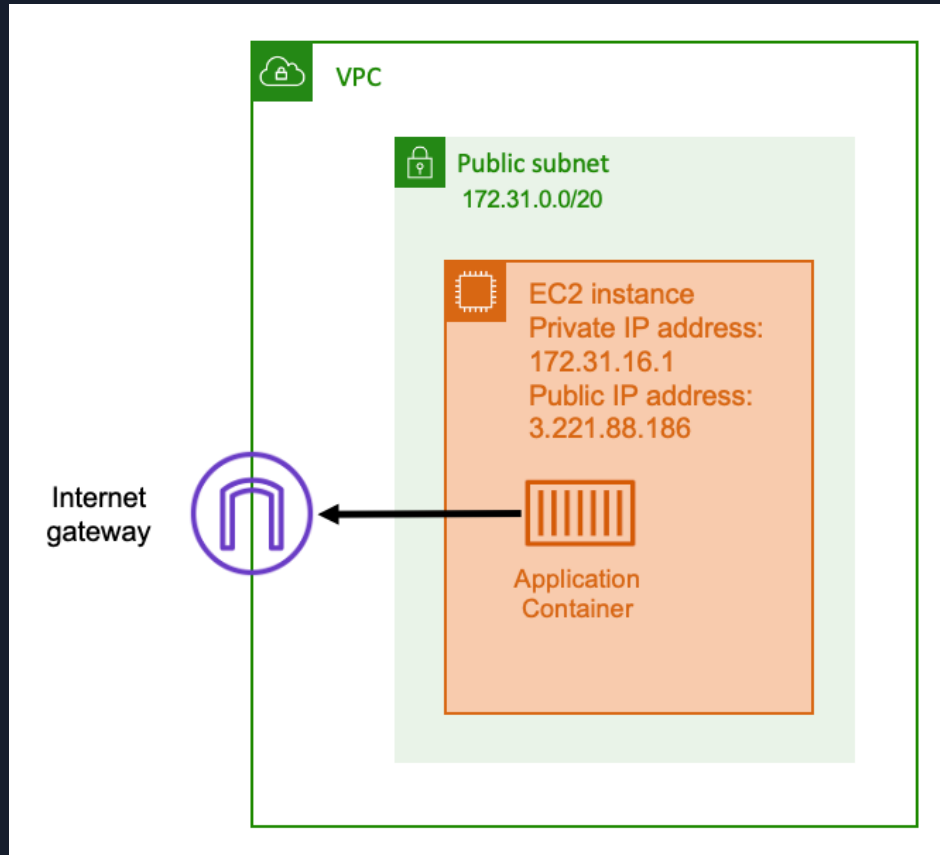
Fargate로 리소스를 보다 세밀하게 제어하여 낭비되는 용량을 최소화합니다.



Amazon ECS 네트워크

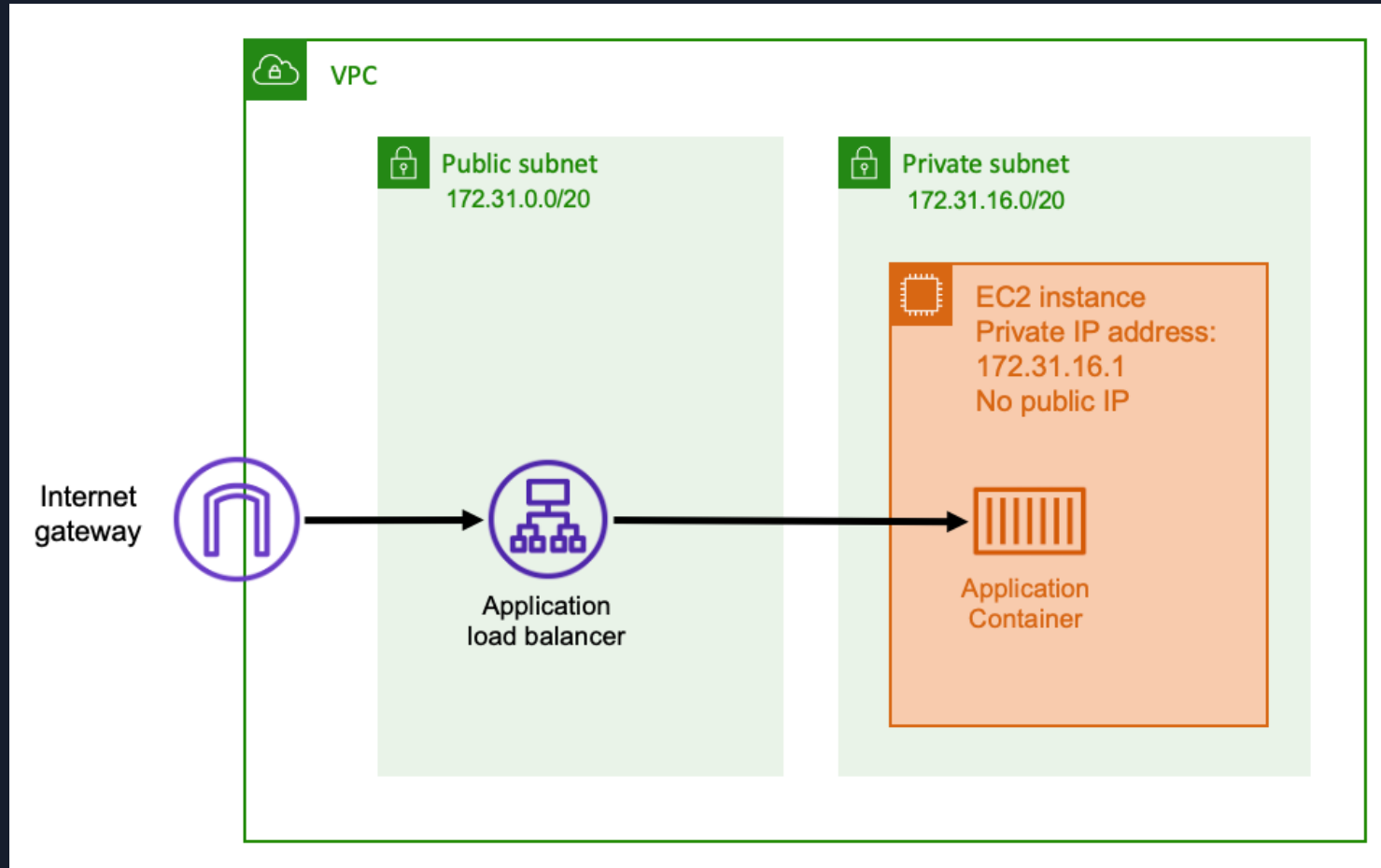
Amazon ECS 네트워크

NETWORKING – OUTBOUND (PUBLIC / PRIVATE SUBNET)



Amazon ECS 네트워크

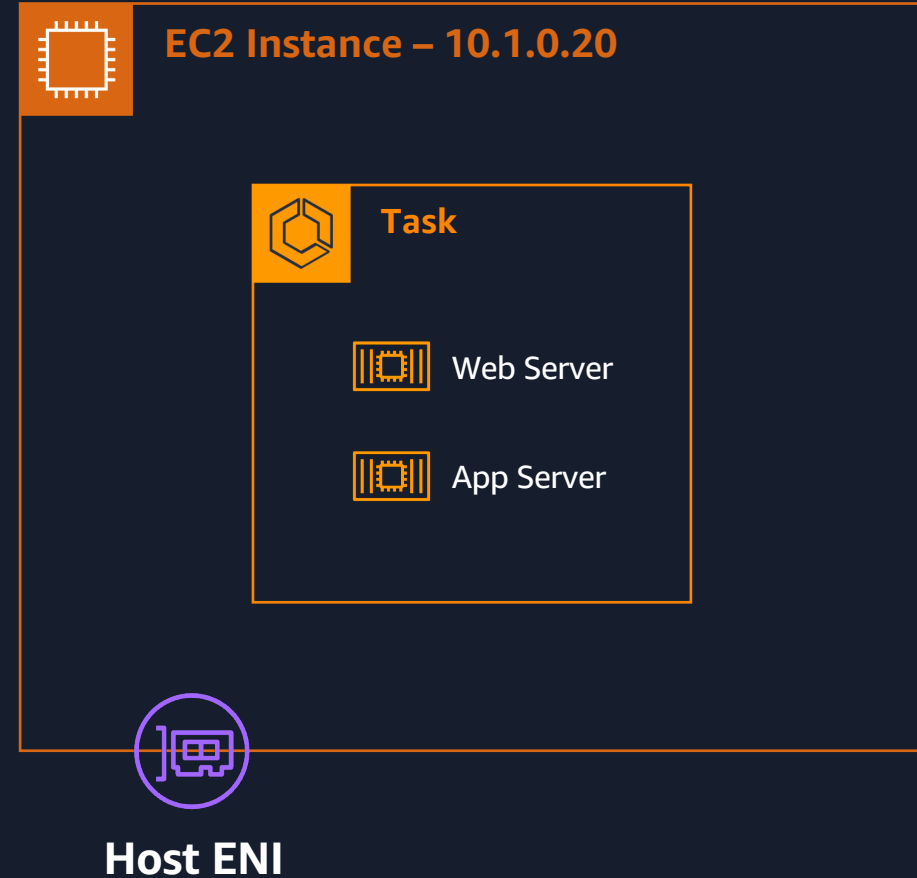
NETWORKING – INBOUND (ALB)



Amazon ECS 네트워크

NETWORKING - NONE

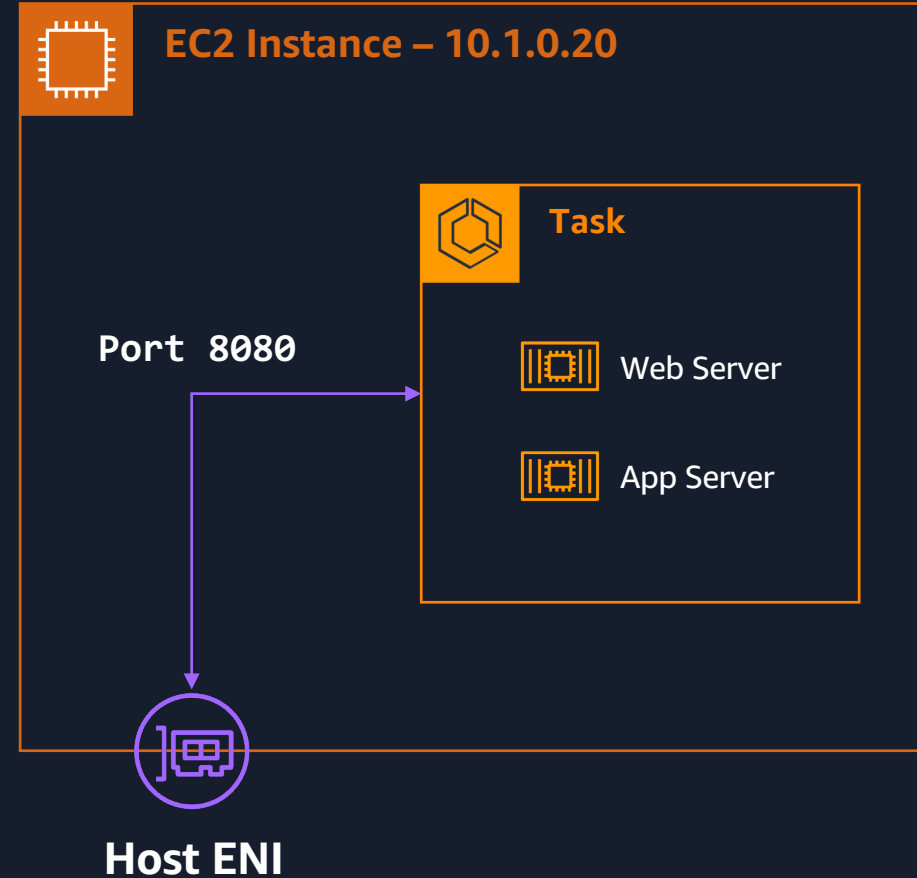
None으로 네트워크 모드가 설정되면
작업의 컨테이너에 외부 연결이 없으며
컨테이너 정의에서 포트 매핑을 지정할 수
없음.



Amazon ECS 네트워크

NETWORKING - HOST

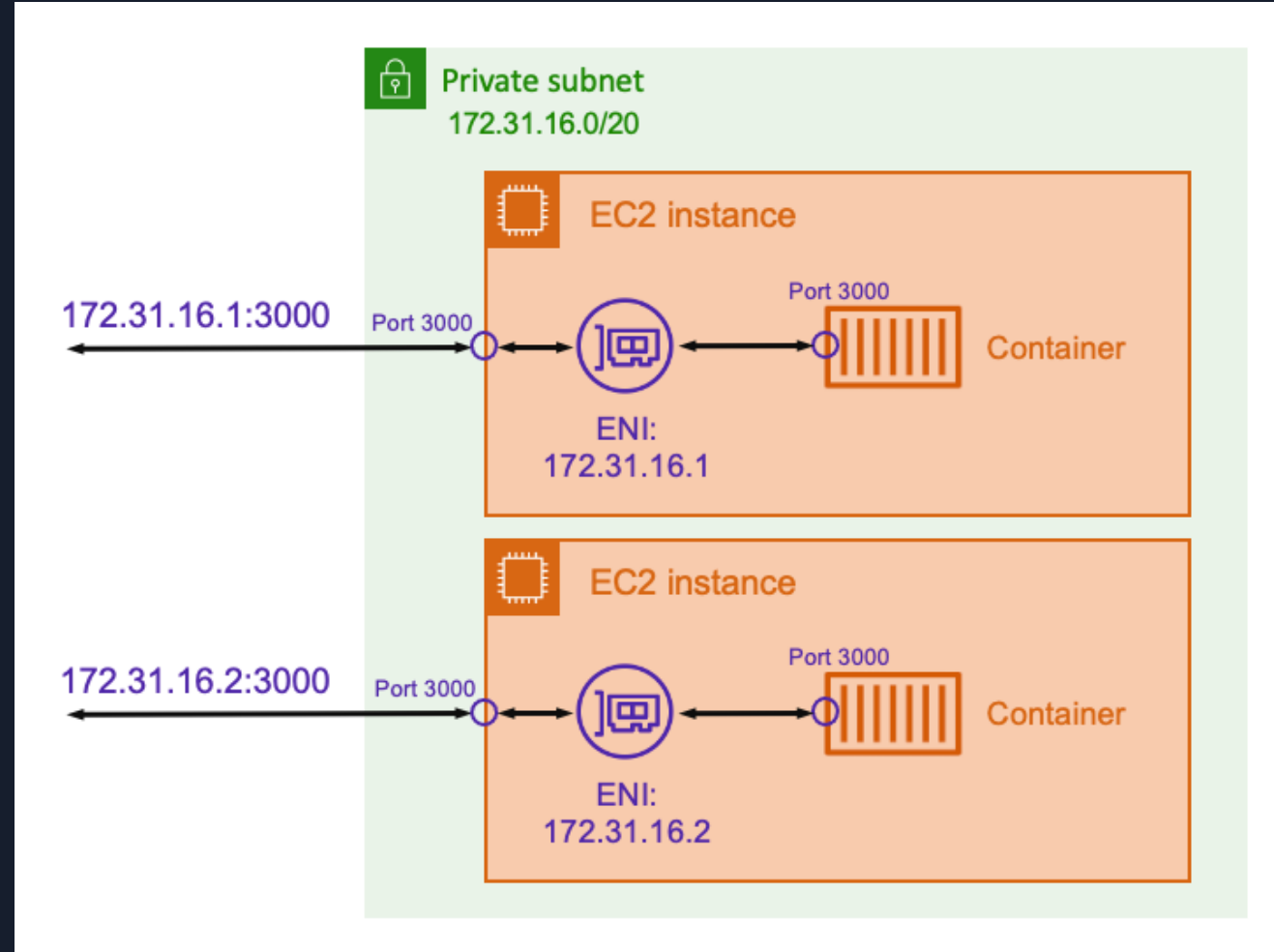
네트워크 모드가 **host**인 경우, 작업은 도커의 빌트인 가상 네트워크를 우회하고 컨테이너 포트를 EC2 인스턴스의 네트워크 인터페이스에 직접 맵핑함. 이 모드에서는 포트 맵핑이 사용될 때 단일 컨테이너 인스턴스에서 동일한 작업의 여러 인스턴스화를 실행할 수 없음.



Amazon ECS 네트워크

NETWORKING - HOST

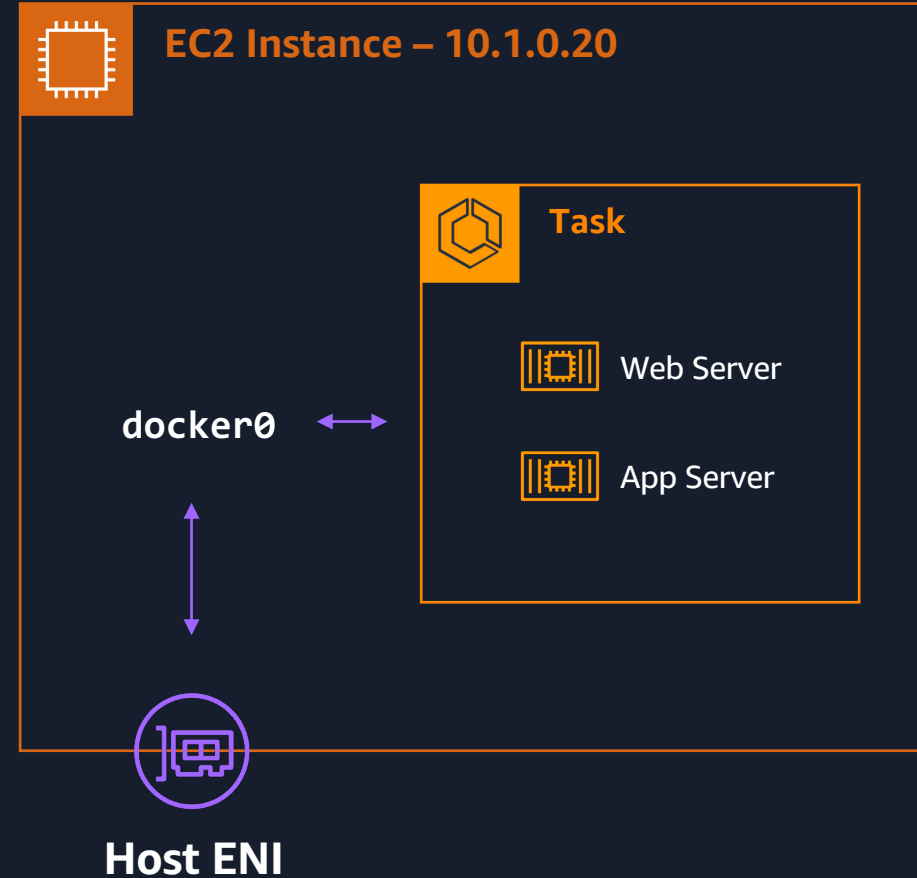
네트워크 모드가 **host**인 경우, 작업은 도커의 빌트인 가상 네트워크를 우회하고 컨테이너 포트를 EC2 인스턴스의 네트워크 인터페이스에 직접 맵핑함. 이 모드에서는 포트 맵핑이 사용될 때 단일 컨테이너 인스턴스에서 동일한 작업의 여러 인스턴스화를 실행할 수 없음.



Amazon ECS 네트워크

NETWORKING - BRIDGE

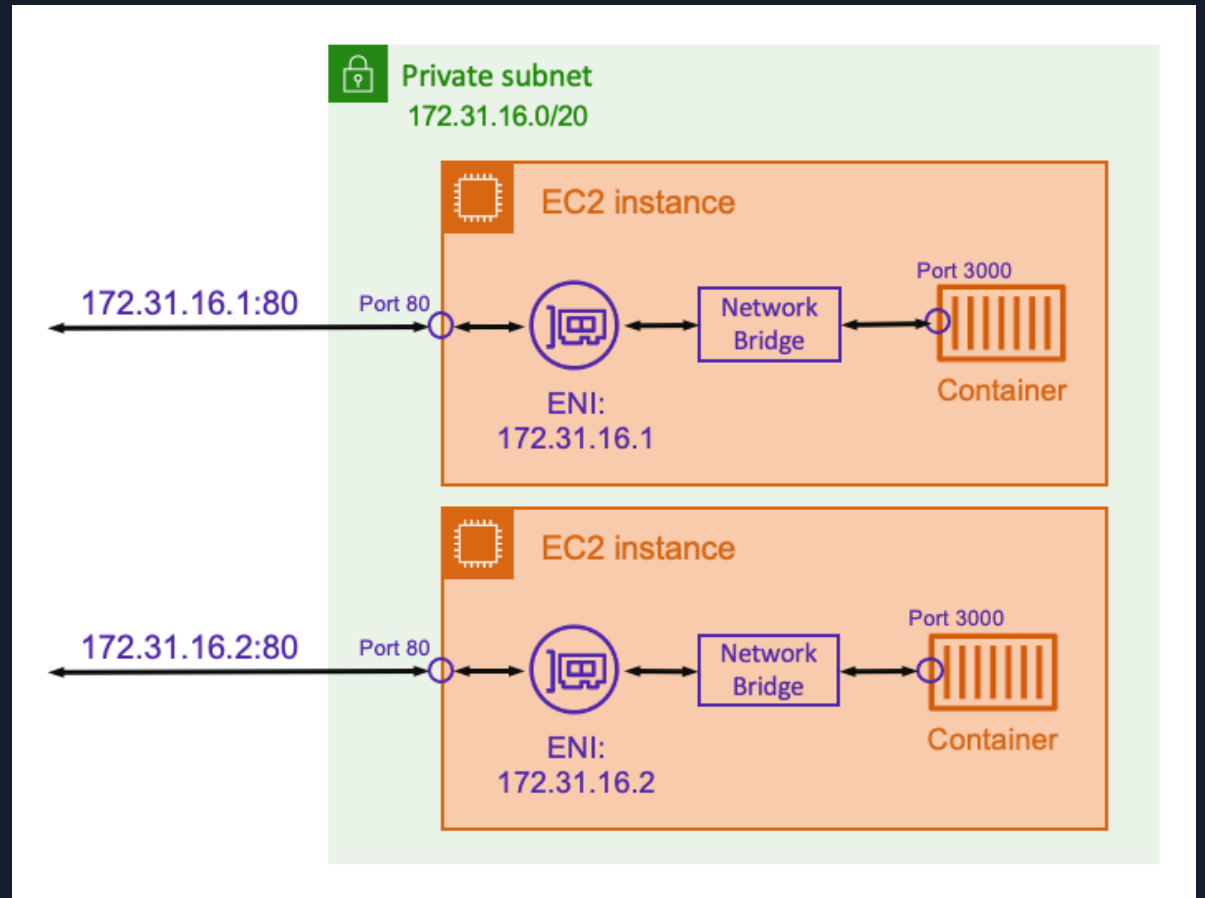
Bridge로 네트워크 모드가 설정되면
작업은 각 컨테이너 인스턴스 내에
실행되는 Docker의 빌트인 가상
네트워크를 활용함.



Amazon ECS 네트워크

NETWORKING - BRIDGE

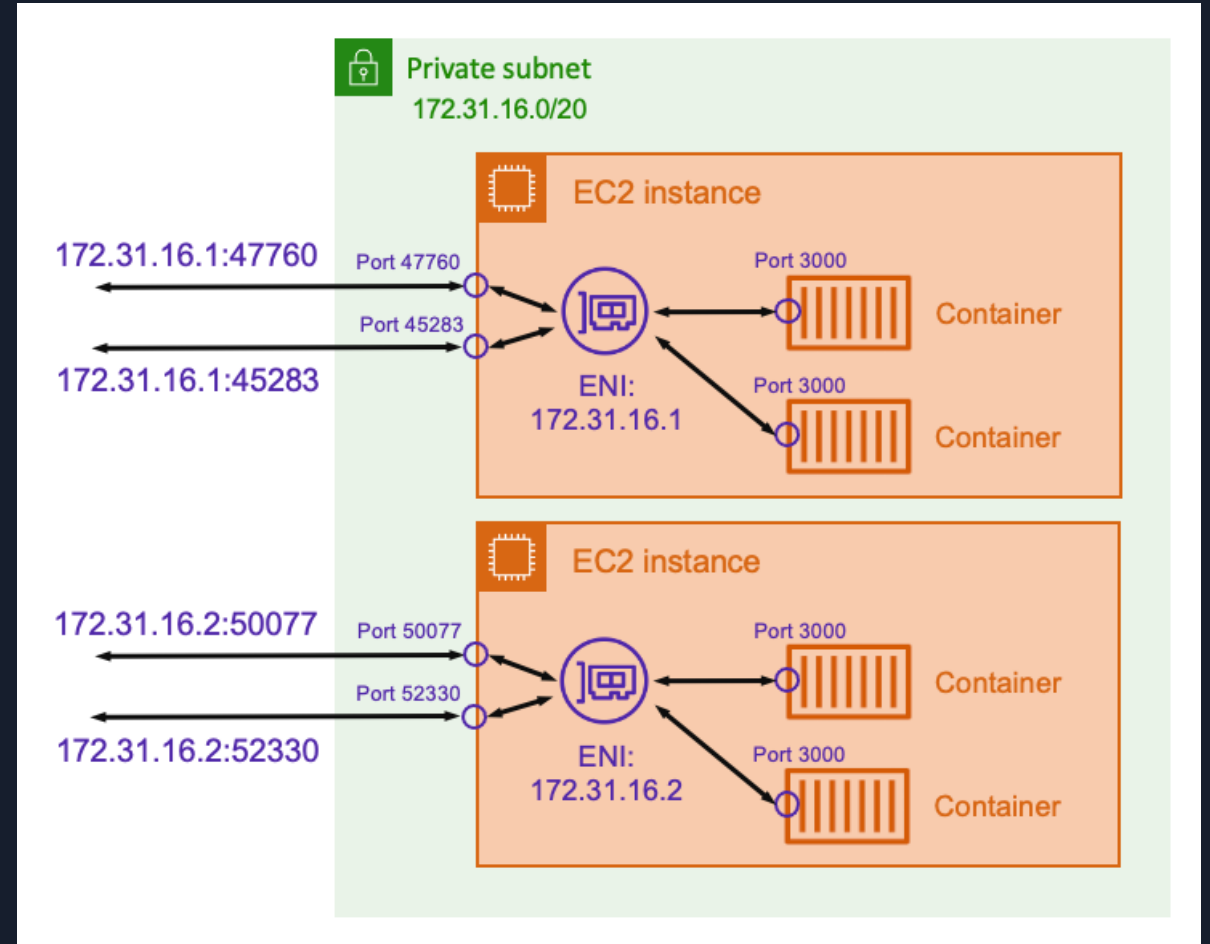
Bridge로 네트워크 모드가 설정되면
작업은 각 컨테이너 인스턴스 내에
실행되는 Docker의 빌트인 가상
네트워크를 활용함.



Amazon ECS 네트워크

NETWORKING - BRIDGE

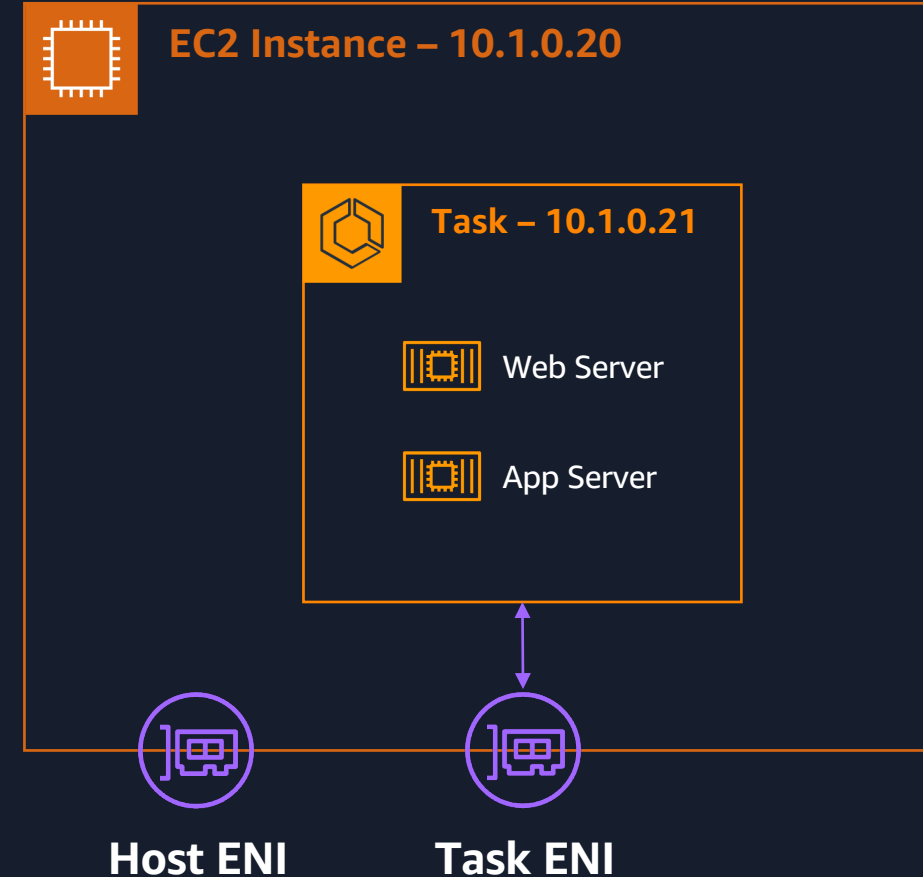
Bridge로 네트워크 모드가 설정되면
작업은 각 컨테이너 인스턴스 내에
실행되는 Docker의 빌트인 가상
네트워크를 활용함.



Amazon ECS 네트워크

NETWORKING - AWSVPC

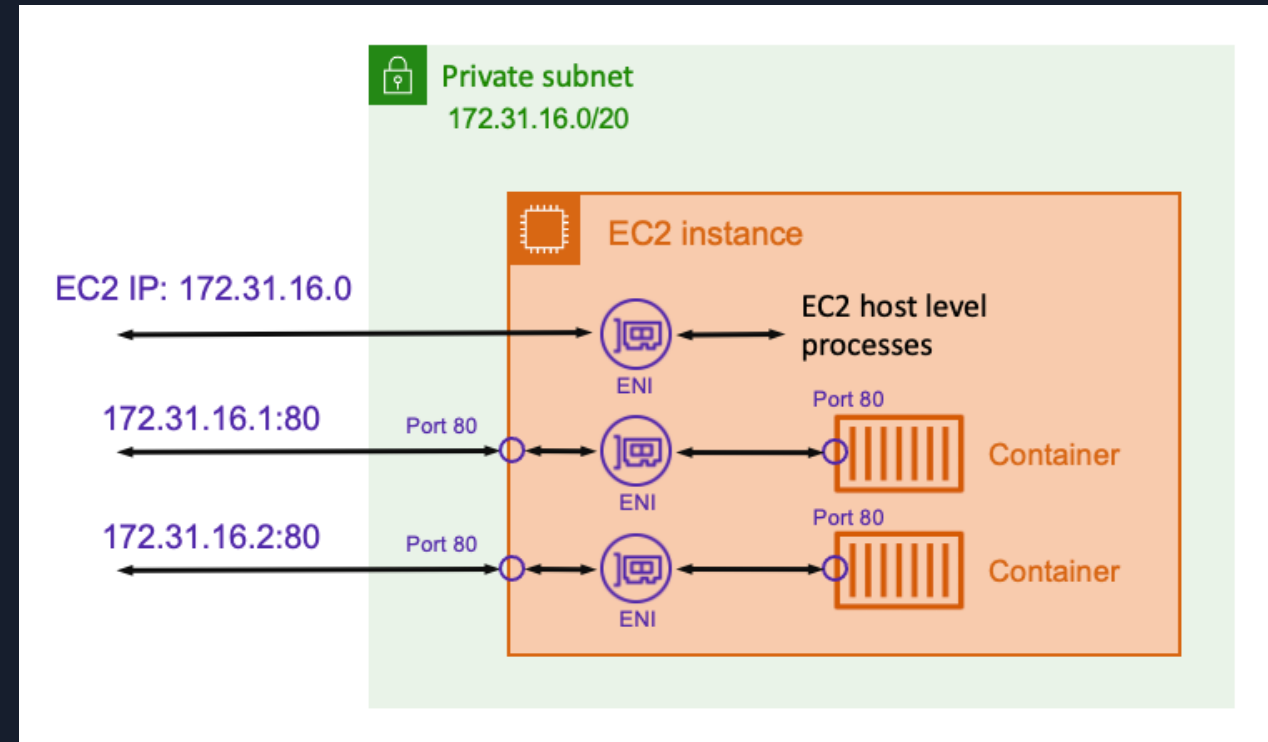
- awsvpc 네트워크 모드는 task 당 elastic network interface를 할당
- 작업에 Amazon EC2 인스턴스와 동일한 네트워킹 속성을 적용할 수 있음
- Fargate는 awsvpc 네트워크 모드에서만 지원됨
- 지원되는 인스턴스
 - Amazon ECS-optimized AMI
 - ecs-init package가 포함된 Amazon Linux



Amazon ECS 네트워크

NETWORKING - AWSVPC

- awsvpc 네트워크 모드는 task 당 elastic network interface를 할당
- 작업에 Amazon EC2 인스턴스와 동일한 네트워킹 속성을 적용할 수 있음
- Fargate는 awsvpc 네트워크 모드에서만 지원됨
- 지원되는 인스턴스
 - Amazon ECS-optimized AMI
 - ecs-init package가 포함된 Amazon Linux

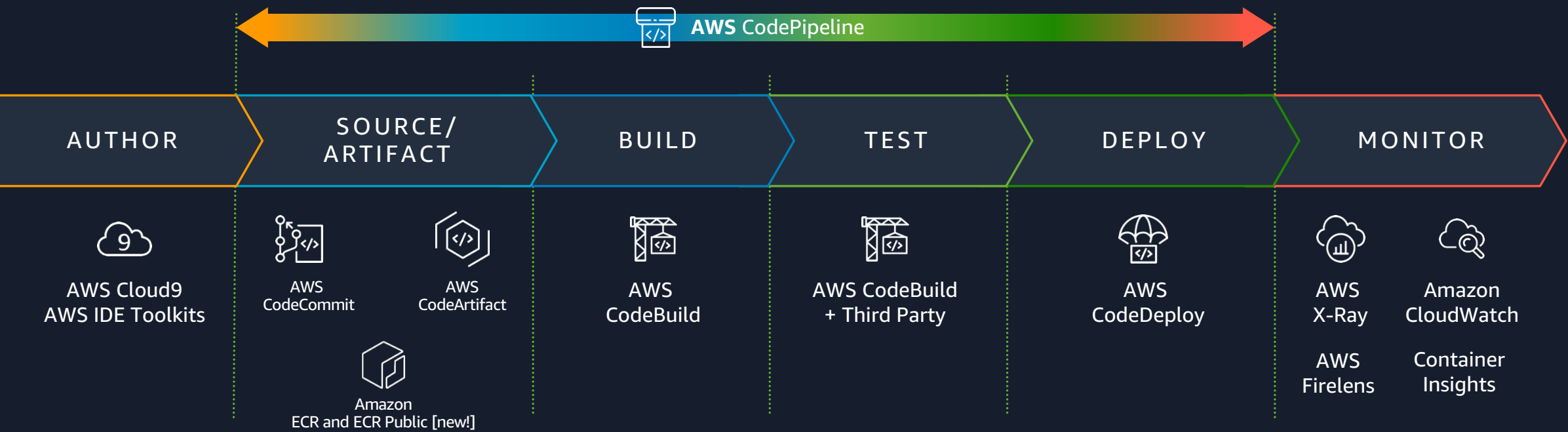


AWS Developer Tool



AWS Developer Tool


AUTOMATE DEPLOYMENT WITH AWS DEVELOPER TOOLS



MODEL

 **AWS CloudFormation**

 **AWS Cloud Development Kit**
(CDK, CDK8s, CDK-terraform)

 **AWS Amplify**

 **AWS Copilot**

Docker Compose

AWS Developer Tool

AWS CODEPIPELINE



- 완전 관리형 지속적 전달(continuous delivery) 서비스
- 릴리즈 프로세스 모델링 및 시각화
- 소스 코드 변경 시, 파이프라인(빌드-테스트-배포 프로세스) 자동 트리거
- Third-party 툴과의 긴밀한 연동

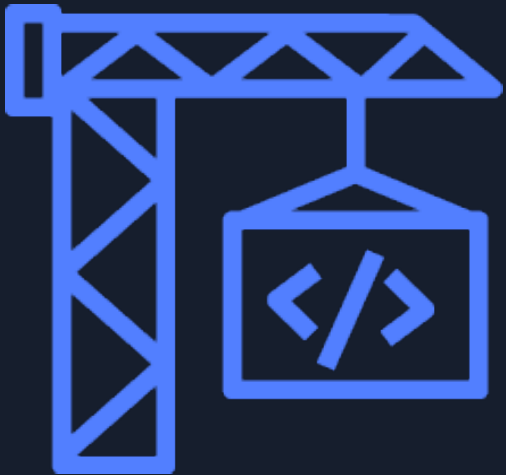
AWS Developer Tool

AWS CODEPIPELINE

- **Source Provider : 소스 코드 저장 위치**
 - branch: AWS CodeCommit, GitHub, Bitbucket
 - object/folder: Amazon S3
 - Docker image: Amazon ECR
- **Builder Provider : 애플리케이션 빌드 방법**
 - AWS CodeBuild, Jenkins
- **Deploy Provider : 애플리케이션 배포 방법**
 - Amazon EC2: AWS CodeDeploy, AWS Elastic Beanstalk, AWS OpsWorks Stacks
 - Containers: AWS CodeDeploy, Amazon ECS, Amazon ECS(blue/green deployment), AWS Fargate
 - Serverless: AWS CodeDeploy, AWS CloudFormation/AWS SAM, AWS Lambda, AWS Step Functions

AWS Developer Tool

AWS CODEBUILD



- 소스 코드를 컴파일하는 단계부터 테스트 실행 후, 소프트웨어 패키지를 개발하여 배포하는 단계까지 마칠 수 있는 완전 관리형 빌드 서비스
- 빌드 볼륨에 따라 자동으로 확장 및 축소
- 빌드를 완료할 때까지 걸리는 시간(분)을 기준으로 과금
- 일관되고 불변하는 환경을 위해 격리된 빌드 컨테이너
- 모든 공식 CodeBuild 이미지는 Docker와 AWS CLI를 포함
- 빌드 환경 커스터마이징 가능

AWS Developer Tool

AWS CODEBUILD – BUILDSPEC FILE

version: 0.2

phases:

pre_build:

commands:

- echo Logging in to Amazon ECR...
- aws --version
- aws ecr get-login-password --region \$AWS_DEFAULT_REGION | docker login --username AWS --password-

stdin 012345678910.dkr.ecr.us-west-2.amazonaws.com

- REPOSITORY_URI=012345678910.dkr.ecr.us-west-2.amazonaws.com/hello-world
- COMMIT_HASH=\$(echo \$CODEBUILD_RESOLVED_SOURCE_VERSION | cut -c 1-7)
- IMAGE_TAG=\${COMMIT_HASH:=latest}

build:

commands:

- echo Build started on `date`
- echo Building the Docker image...
- docker build -t \$REPOSITORY_URI:latest .
- docker tag \$REPOSITORY_URI:latest \$REPOSITORY_URI:\$IMAGE_TAG

post_build:

commands:

- echo Build completed on `date`
- echo Pushing the Docker images...
- docker push \$REPOSITORY_URI:latest
- docker push \$REPOSITORY_URI:\$IMAGE_TAG
- echo Writing image definitions file...
- printf ' [{"name":"hello-world","imageUri":"%s"}] ' \$REPOSITORY_URI:\$IMAGE_TAG > imagedefinitions.json

artifacts:

files: imagedefinitions.json

AWS Developer Tool

AWS CODEDEPLOY



- 코드 배포를 자동화하는 완전 관리형 배포 서비스
- 복잡한 애플리케이션 업데이트 작업을 처리
- 배포 중, 다운타임 최소화
- 배포 중, 오류 감지 시, 자동으로 롤백
- Amazon EC2, AWS Fargate, AWS Lambda, Amazon ECS, 온프레미스 서버에 배포

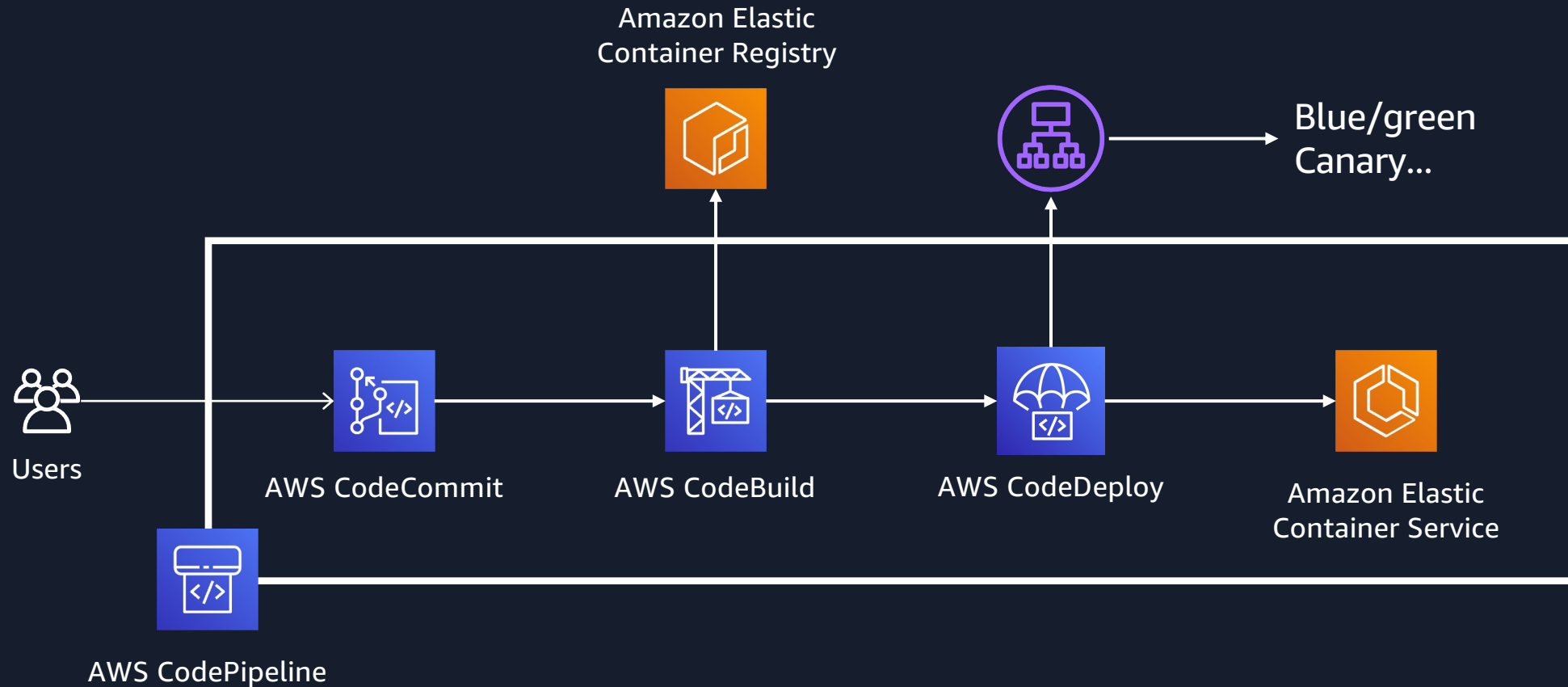
AWS Developer Tool

AWS CODEDEPLOY – APPSPEC FILE

```
version: 0.0
Resources:
  - TargetService:
      Type: AWS::ECS::Service
      Properties:
        TaskDefinition: "arn:aws:ecs:us-east-1:111222333444:task-definition/my-task-definition-family-name:1"
        LoadBalancerInfo:
          ContainerName: "SampleApplicationName"
          ContainerPort: 80
# Optional properties
        PlatformVersion: "LATEST"
        NetworkConfiguration:
          AwsvpcConfiguration:
            Subnets: ["subnet-1234abcd", "subnet-5678abcd"]
            SecurityGroups: ["sg-12345678"]
            AssignPublicIp: "ENABLED"
        CapacityProviderStrategy:
          - Base: 1
            CapacityProvider: "FARGATE_SPOT"
            weight: 2
          - Base: 0
            CapacityProvider: "FARGATE"
            weight: 1
Hooks:
  - BeforeInstall: "LambdaFunctionToValidateBeforeInstall"
  - AfterInstall: "LambdaFunctionToValidateAfterInstall"
  - AfterAllowTestTraffic: "LambdaFunctionToValidateAfterTestTrafficStarts"
  - BeforeAllowTraffic: "LambdaFunctionToValidateBeforeAllowingProductionTraffic"
  - AfterAllowTraffic: "LambdaFunctionToValidateAfterAllowingProductionTraffic"
```

AWS Developer Tool

컨테이너 배포 자동화



실습 시작 전 준비 사항

AWS 계정으로 시작

1. 실습 전 계정을 꼭 신청해주세요 : <https://portal.aws.amazon.com/billing/signup#/start>
2. AWS 계정이 없으신 경우, 행사 참여 전에 미리 AWS 계정 생성 가이드를 확인하시고 AWS 계정을 생성해 주시길 바랍니다.

*AWS 계정 생성 가이드: <https://aws.amazon.com/ko/premiumsupport/knowledge-center/create-and-activate-aws-account/>

3. 검증된 호환성을 위하여 실습 시 사용할 웹 브라우저는 Mozilla Firefox 또는 Google Chrome Browser로 진행 부탁드립니다.

실습 마무리 및 설문 참여 방법

- 실습이 모두 끝난 후에는 **자원 삭제**를 잊지 마세요. 직접 준비하신 AWS 계정으로 실습을 진행하신 고객 분들의 경우, 가이드에 따라 자원 삭제를 진행하셔야 합니다. 또한, 기존에 사용하시던 자원이 있으신 고객 분들의 경우, **오늘 생성한 자원만 삭제**하는 것에 주의 부탁드립니다.
- **가이드:** (세션별 제공)
- 마지막으로 세션이 끝난 후, **GoToWebinar 창을 종료하면 설문 조사 창**이 나옵니다.
이때, **설문 조사를 진행해 주셔야 AWS 크레딧**(1인당 \$50 크레딧, 전체 세션당 1회 제공)을 제공받으실 수 있습니다.

AWS는 고객 피드백을 기반으로 의사 결정을 수행하며 이러한 피드백은 추후에 진행할 세션 방향을 결정합니다.

더 나은 세션을 위하여 여러분들의 소중한 의견을 부탁드립니다.

감사합니다.

크레딧 안내

- AWS 계정으로 시작하실 경우, **금일 실습에서 발생하는 비용은 당월 과금이 되는 점** 미리 확인 부탁드립니다.
- 웨비나 종료 후 **설문 조사에 참여해주신 분들께는 AWS 크레딧 바우처** (1인당 \$50 USD 크레딧, 전체 세션당 1회 제공)를 드립니다.
- 해당 **AWS 크레딧**은 등록하신 이메일 계정으로 **행사 종료 후 1개월 내** 발송 드릴 예정이며, 전달 받은 AWS 크레딧은 바로 사용 가능합니다.

감사 메일 & 참석 증명서

- AWS TechCamp 온라인 세션에 참석해 주신 분들께 행사 종료 후 1개월 내 감사메일과 참석 증명서가 순차 발송됩니다.
- 등록 진행 후 참석하지 않으실 경우 별도 메일 및 증명서는 발급되지 않습니다.

감사 메일 예시

**AWS TechCamp 온라인 세미나에
참석해 주셔서 감사합니다.**

AWS Builders Korea Program에 참석하고 피드백을 공유해주셔서
감사드립니다. 세미나 자료는 아래 링크를 통해 확인하실 수 있습니다.

자료 확인하기

참석 증명서 예시

참석 증명서

AWS TechCamp 온라인 세미나에 참석해 주셔서 감사합니다.

홍길동

2024년 3월 26일 - 3월 28일



Thank you!

Jaemin Jung

jungjm@amazon.com



서베이에 참여해 여러분의
의견을 공유해 주세요!