

AWS 200: DevOps 서비스

AWS 솔루션즈 아키텍트 이민우

2020.07.15



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

- **DevOps** 개념
- **DevOps** 를 위한 도구
 - 통합개발환경 (**IDE**)
 - **Code Repository**
 - 지속적 통합 (**CI**)
 - 코드로서의 인프라 (**IaC**)
 - **Deployment**
 - **Debugging**
 - 파이프라인 관리
- 데모(실습)

DevOps 개념



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

DevOps란 무엇인가?



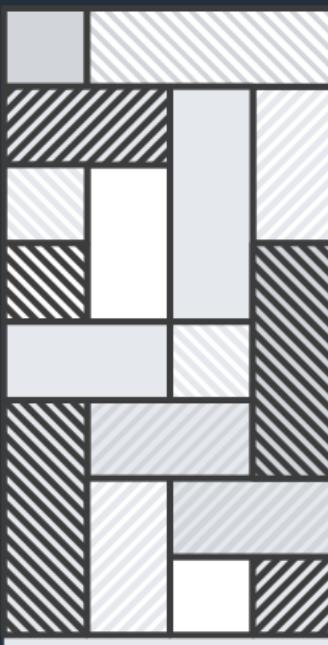
문화적 철학
관행
도구들(**Tools**)

책임 공유
Ownership
가시성 및 의사소통

DevOps 관행

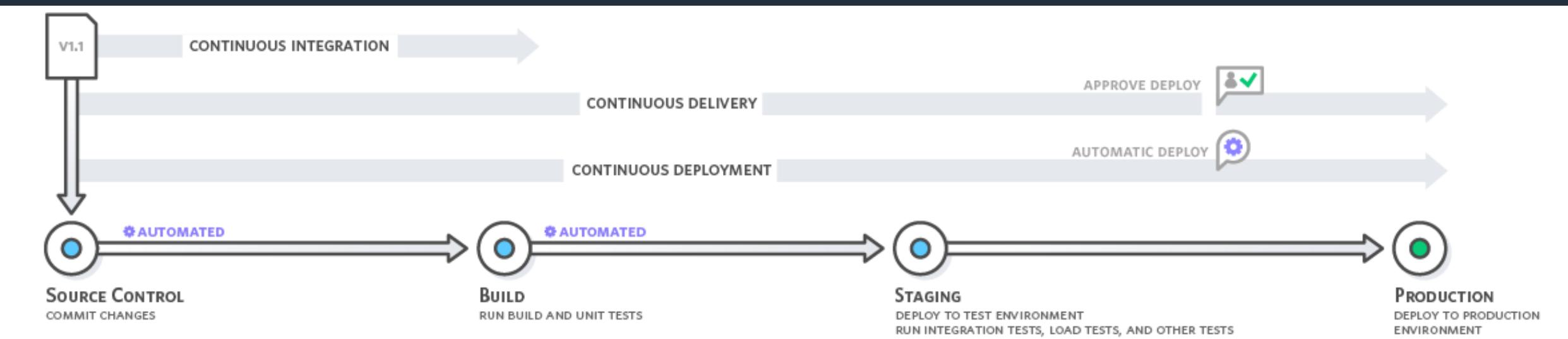
マイ크로서비스 (**Microservices**)

- **Monolithic** 어플리케이션 아키텍처에서 여러개의 개별 서비스로 전환



DevOps 관행

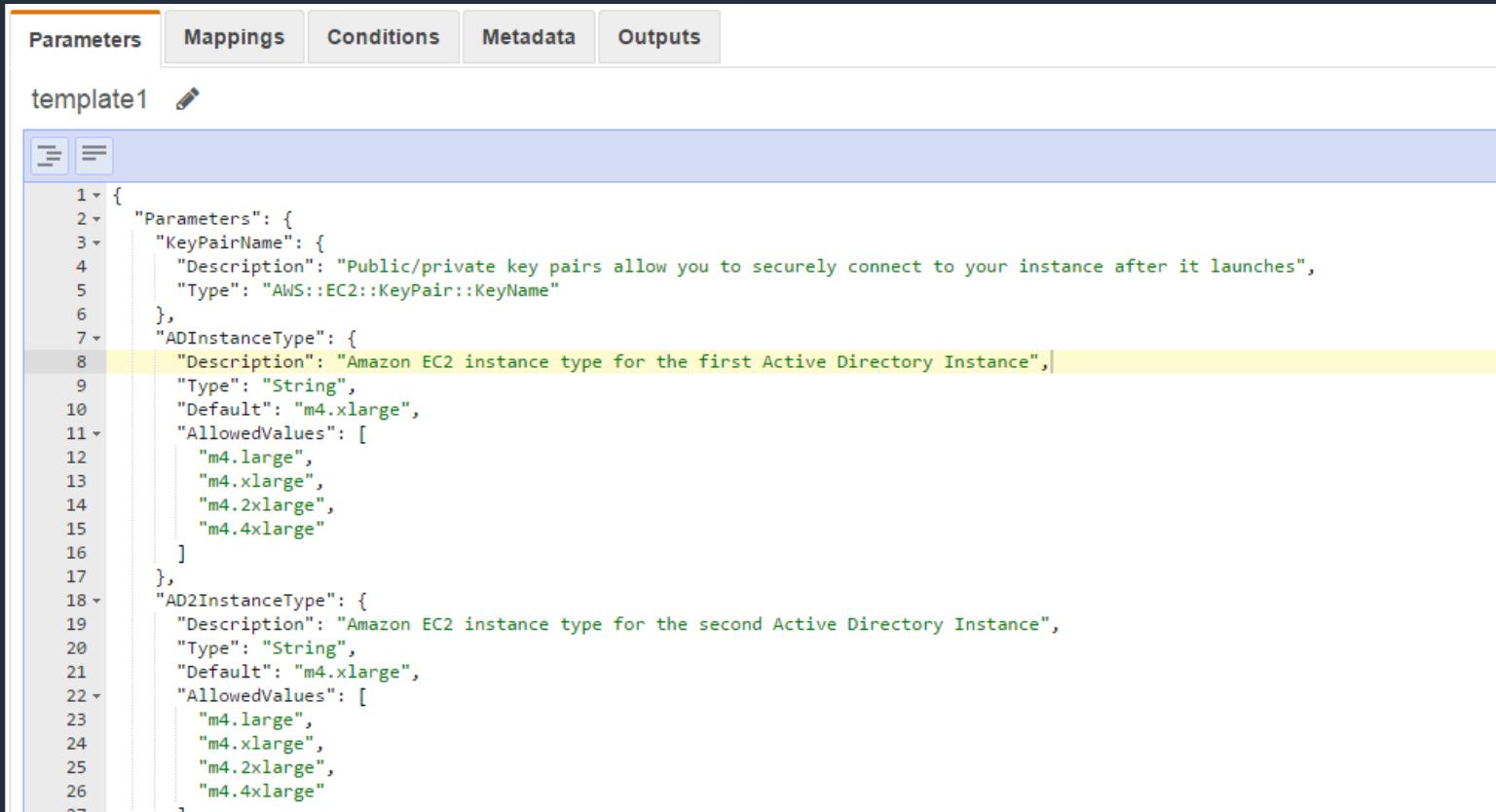
- 지속적 통합 (**Continuous integration**)
- 지속적 전달/배포 (**Continuous delivery & deployment**)



DevOps 관행

Infrastructure as code

- 코드를 사용하여 AWS 리소스를 모델링



The screenshot shows the AWS CloudFormation template editor with the 'Parameters' tab selected. The template is named 'template1'. It contains two parameters: 'ADInstanceType' and 'AD2InstanceType'. Both parameters are of type 'String' and have a default value of 'm4.xlarge'. They also have allowed values: 'm4.large', 'm4.xlarge', 'm4.2xlarge', and 'm4.4xlarge'. The 'ADInstanceType' parameter has a detailed description: "Amazon EC2 instance type for the first Active Directory Instance". The 'AD2InstanceType' parameter has a similar description: "Amazon EC2 instance type for the second Active Directory Instance".

```
1: {
2:   "Parameters": {
3:     "KeyPairName": {
4:       "Description": "Public/private key pairs allow you to securely connect to your instance after it launches",
5:       "Type": "AWS::EC2::KeyPair::KeyName"
6:     },
7:     "ADInstanceType": {
8:       "Description": "Amazon EC2 instance type for the first Active Directory Instance",
9:       "Type": "String",
10:      "Default": "m4.xlarge",
11:      "AllowedValues": [
12:        "m4.large",
13:        "m4.xlarge",
14:        "m4.2xlarge",
15:        "m4.4xlarge"
16:      ]
17:    },
18:    "AD2InstanceType": {
19:      "Description": "Amazon EC2 instance type for the second Active Directory Instance",
20:      "Type": "String",
21:      "Default": "m4.xlarge",
22:      "AllowedValues": [
23:        "m4.large",
24:        "m4.xlarge",
25:        "m4.2xlarge",
26:        "m4.4xlarge"
27:      ]
28:    }
29:  }
30: }
```

DevOps 관행

- 모니터링과 로깅
 - 지표(**Metric**) 와 로그들을 추적하고 분석
 - 인프라스트럭처와 어플리케이션의 실시간 성능을 이해



DevOps 의 이점



협업 개선



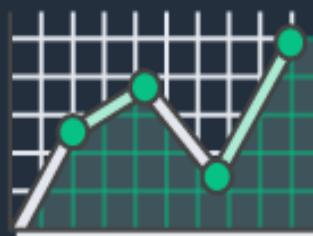
빠른
전달/배포



신뢰성



보안



확장성



속도

숫자로 보는 DevOps 이점

7x

더 적은 변경 실패

106x

더 빠른 **Commit to Deploy**

208x 더
빈번한 배포

2,604x

사고로 부터 더 빠른 복구

Source: The State of DevOps Report 2019



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

릴리스 및 모니터의 5 가지 주요 단계

소스

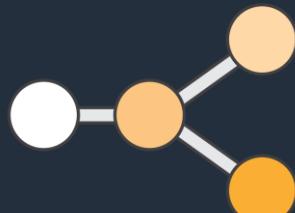
빌드

테스트

배포

모니터

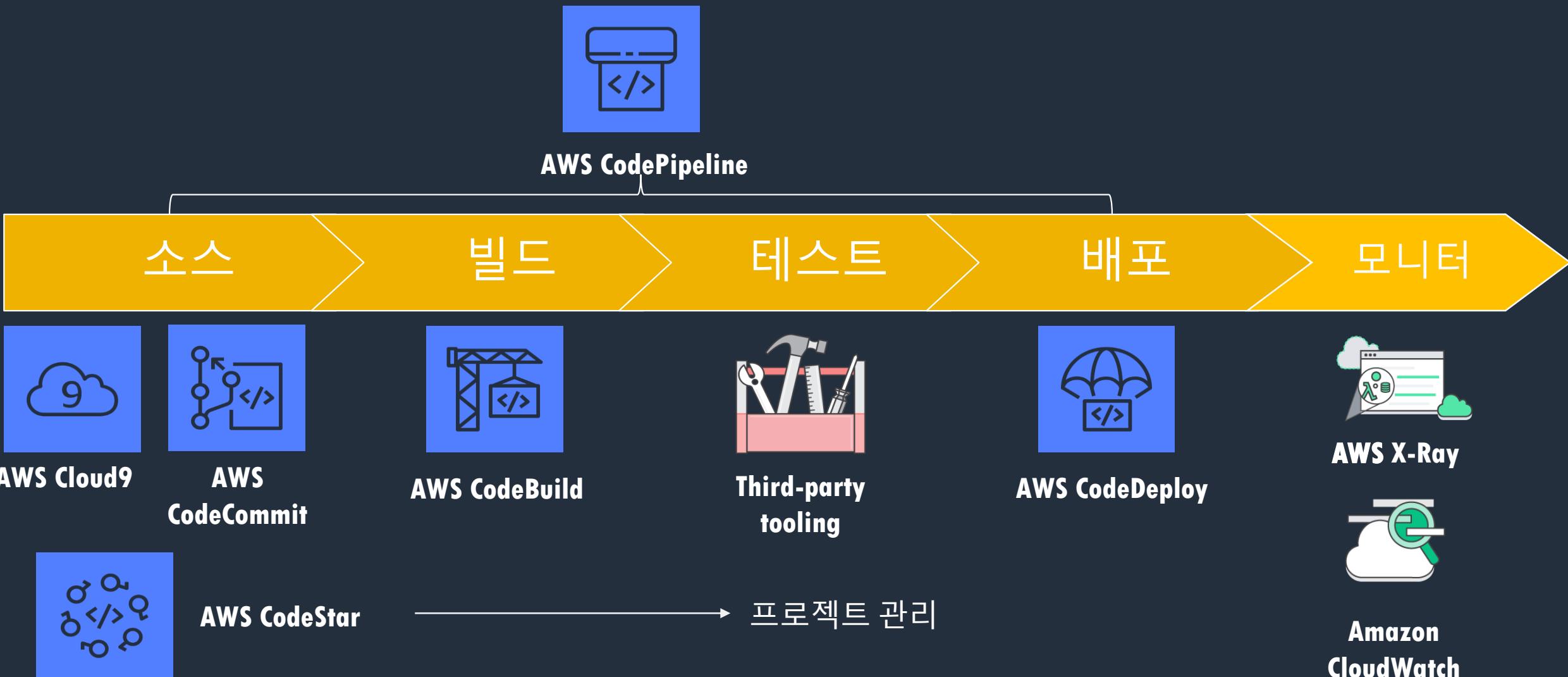
- .java 파일과 같은 소스 코드 체크인
- 신규 코드의 Peer review
- 코드 컴파일
- 단위 테스트
- Style checkers
- Code 지표
- 컨테이너 이미지 생성
- 다른 시스템과의 통합 테스트
- 부하 테스트
- UI 테스트
- 침투 테스트
- 프로덕션 환경에 배포
- 비정상적인 활동이나 오류를 신속하게 감지하기 위해 프로덕션 환경에서 코드 모니터링



릴리스 프로세스 수준



AWS에서의 CI/CD 파이프라인



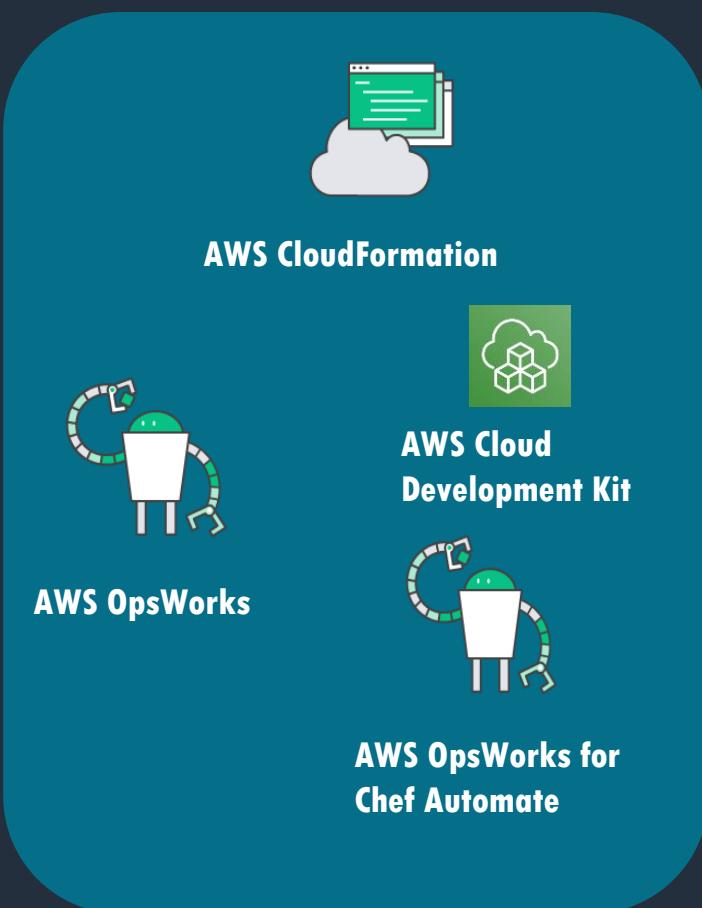
AWS DevOps 포트폴리오

소프트웨어 개발과 지속적 전달

Toolchain



코드로서의 인프라
(IaC)



모니터링 및 로깅



통합개발환경 (IDE)

통합개발환경(IDE) 선택



AWS Toolkit
for PyCharm
Python



AWS Toolkit
for IntelliJ
Java, Python



AWS Toolkit for
Visual Studio Code
.NET, Node



AWS Toolkit for
Visual Studio
.NET

New



AWS Toolkit
for Webstorm
Node.js



AWS Toolkit
for Rider
.NET

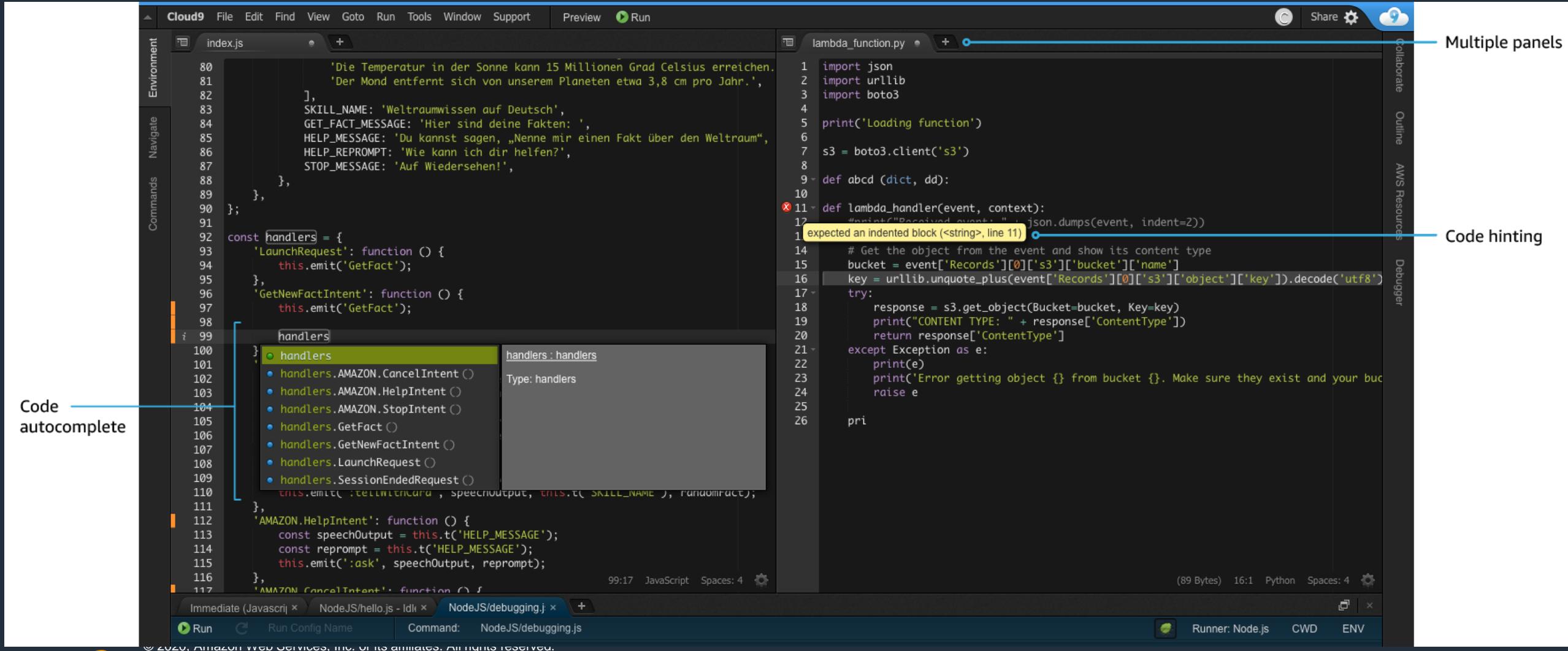
AWS Cloud9

코드 작성, 실행 및 디버깅을 위한 클라우드 **IDE**

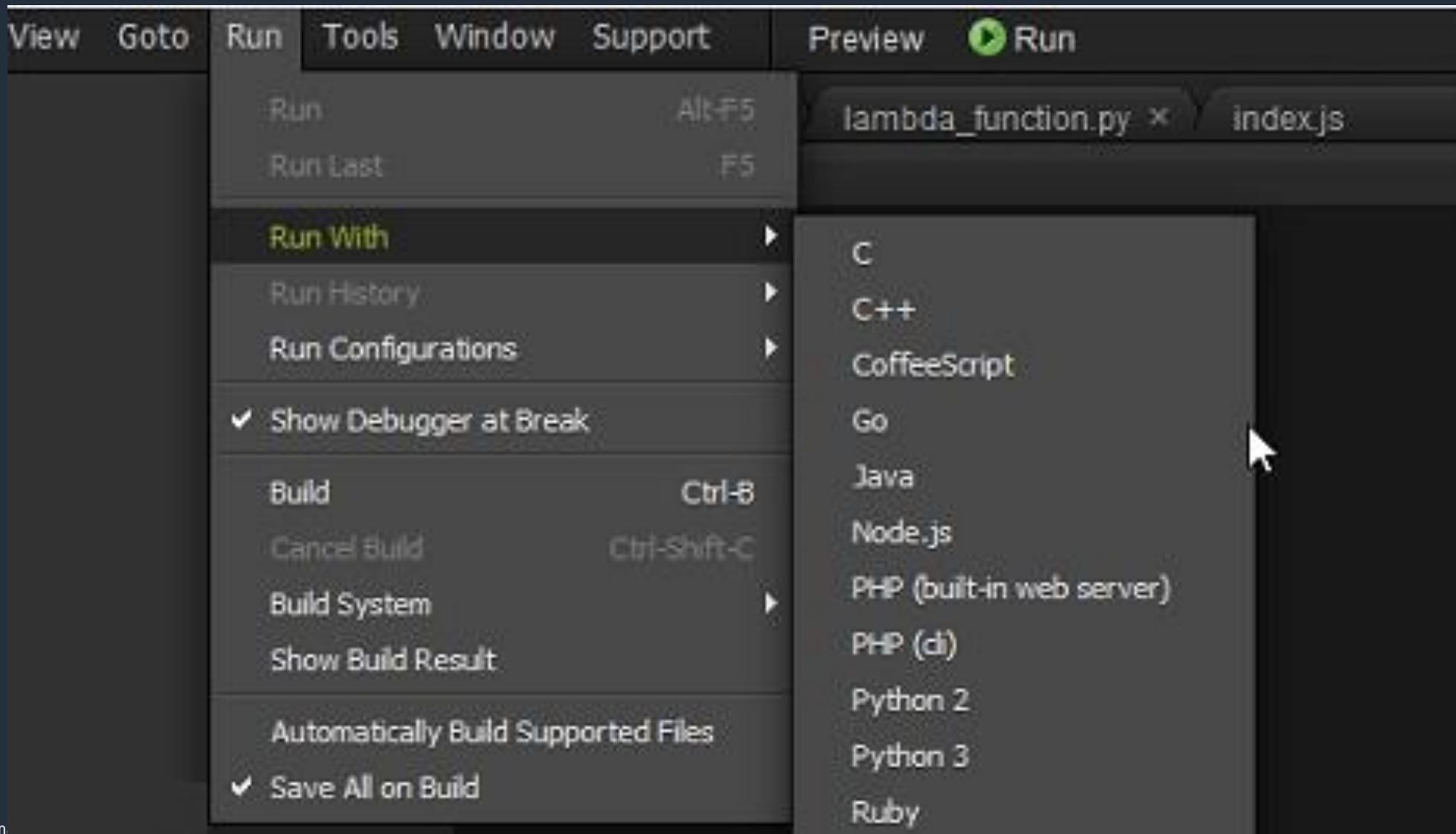
- 브라우저만으로 코딩
- 실시간으로 코딩
- **Serverless** 어플리케이션을 쉽게 구축
- **AWS**에 직접 터미널 엑세스
- 신규 프로젝트를 빠르게 시작



완전 기능화된 편집기



광범위한 Run Time 선택



완전 기능화된 Debugger

The screenshot shows the Cloud9 IDE interface with a debugger panel open. The code editor on the left contains a JavaScript file named 'debugging.js' with the following content:

```
// a javascript program to add two variables
function addVars(first, second) {
  var sum = first + second;
  return sum;
}

var firstVar = 1;
var secondVar = 2;

var total = addVars(firstVar, secondVar);

console.log(total);
```

A blue arrow labeled 'Breakpoint' points to the line 'var secondVar = 2;'. The debugger panel on the right displays the following information:

- Watch Expressions: Step Out (F11 | ⌘ ⌘ ⌘)
- Call Stack: Step Out (F11 | ⌘ ⌘ ⌘)
- Local Variables:

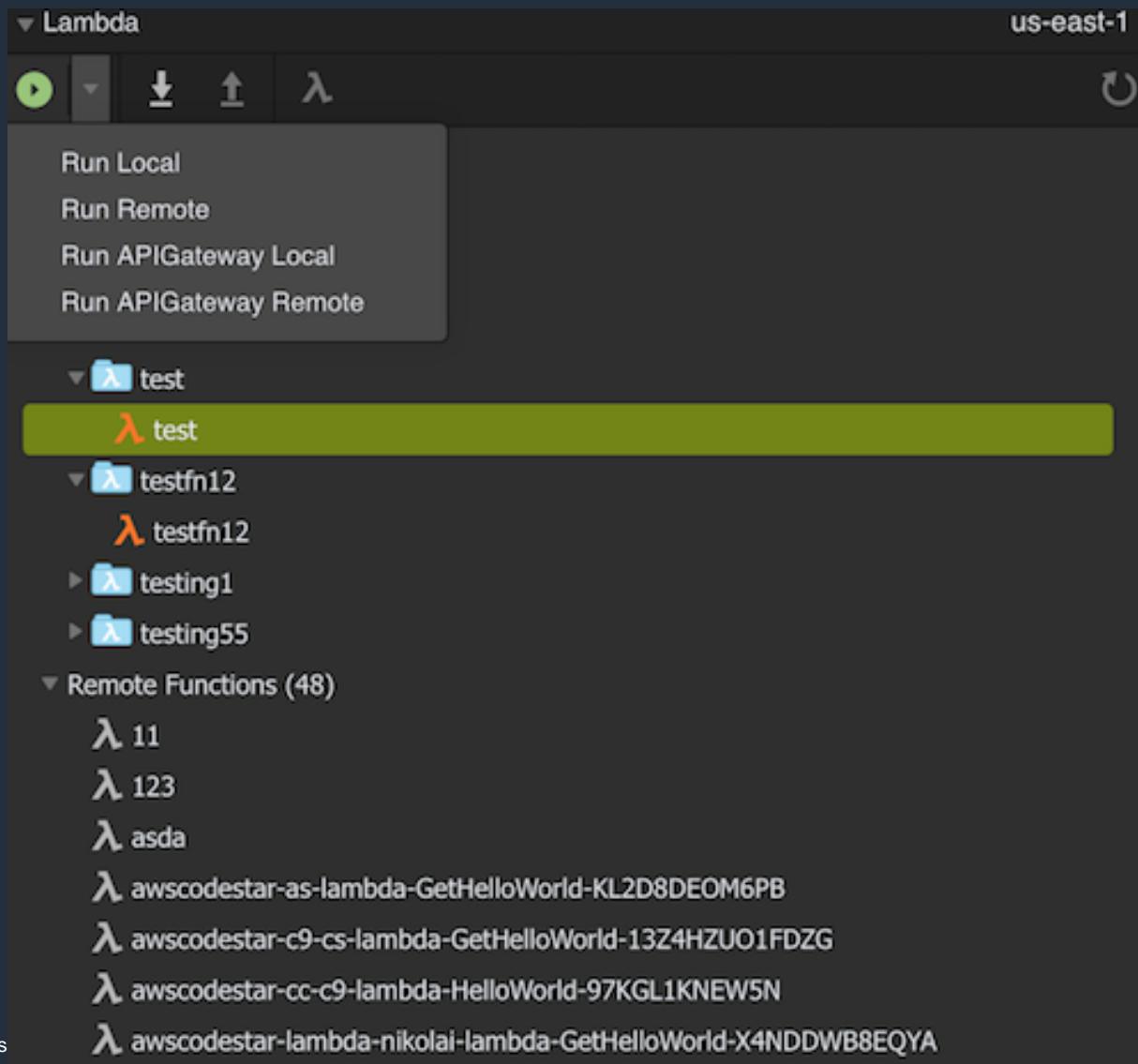
Variable	Type
__dirname	string
__filename	string
addVars	function
exports	object
firstVar	number
module	object
require	function
secondVar	undefined
this	[Object]
total	undefined
Scope	[Object]
Scope	global
- Breakpoints:
 - debugging.js:9 var secondVar = 2;

Annotations on the right side of the screenshot:

- A blue bracket labeled 'Debugging controls' covers the top right area of the interface.
- A blue bracket labeled 'Debugging panel' covers the bottom right area of the interface.



Serverless 개발을 위한 통합 도구



Code Repository



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Code Repository 로 코드 이동

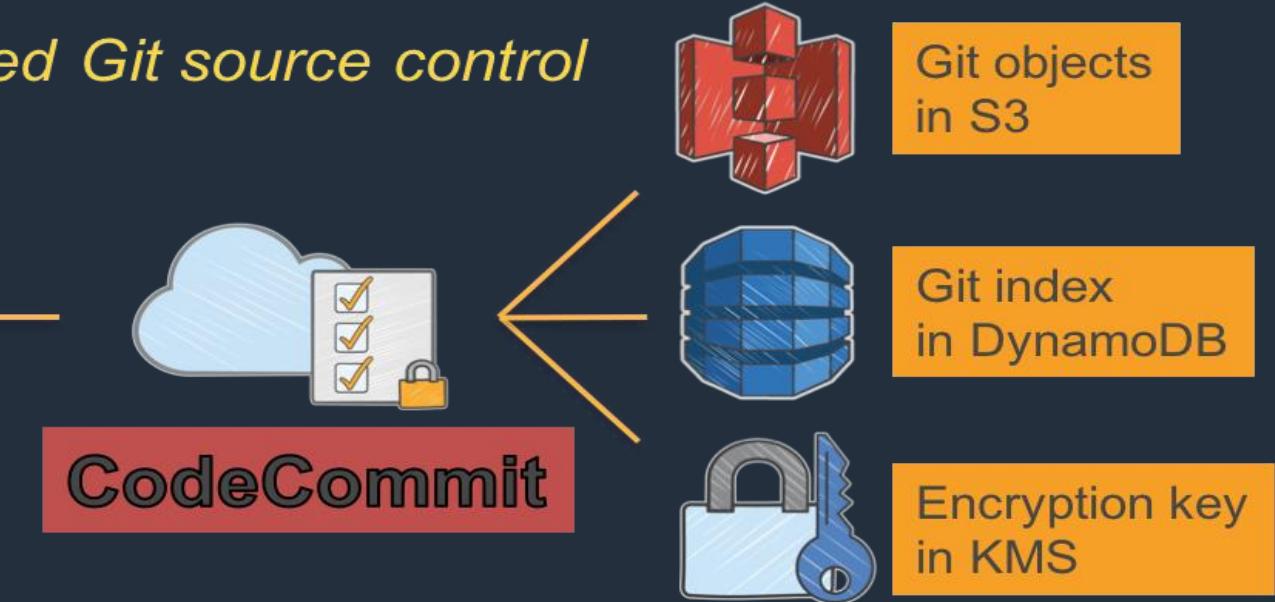


AWS CodeCommit

Secure, scalable, and managed Git source control



SSH or HTTPS



- AZ 간 데이터 이중화
- 저장시 데이터 암호화
- AWS Identity and Access Management (IAM) 와 통합
- Repo 사이즈 제한 없음

CodeCommit & ssh 시작하기

```
$ ssh-keygen
```

SSH Key ID	Uploaded	Status	Actions
APKAEIBAERJR2EXAMPLE	2015-07-21 16:32 PDT	Active	Make Inactive Show SSH Key Delete

```
$ vi ~/.ssh
```

```
Host git-codecommit.*.amazonaws.com
User APKAEiBAERJR2EXAMPLE
identityFile ~/.ssh/codecommit_rsa
```

Git 과 동일한 경험

```
$ git clone https://git-codecommit.us-east-1.amazonaws.com/v1/repos/aws-cli
```

Cloning into 'aws-cli'...

Receiving objects: 100% (16032/16032), 5.55 MiB | 1.25 MiB/s, done.

Resolving deltas: 100% (9900/9900), done.

Checking connectivity... done.

```
$ nano README.rst
```

```
$ git commit -am 'updated README'
```

[master 4fa0318] updated README

1 file changed, 1 insertion(+)

```
$ git push
```

Counting objects: 3, done.

Delta compression using up to 4 threads.

Compressing objects: 100% (3/3), done.

Writing objects: 100% (3/3), 297 bytes | 0 bytes/s, done.

Total 3 (delta 2), reused 0 (delta 0)

remote:

To https://git-codecommit.us-east-1.amazonaws.com/v1/repos/aws-cli

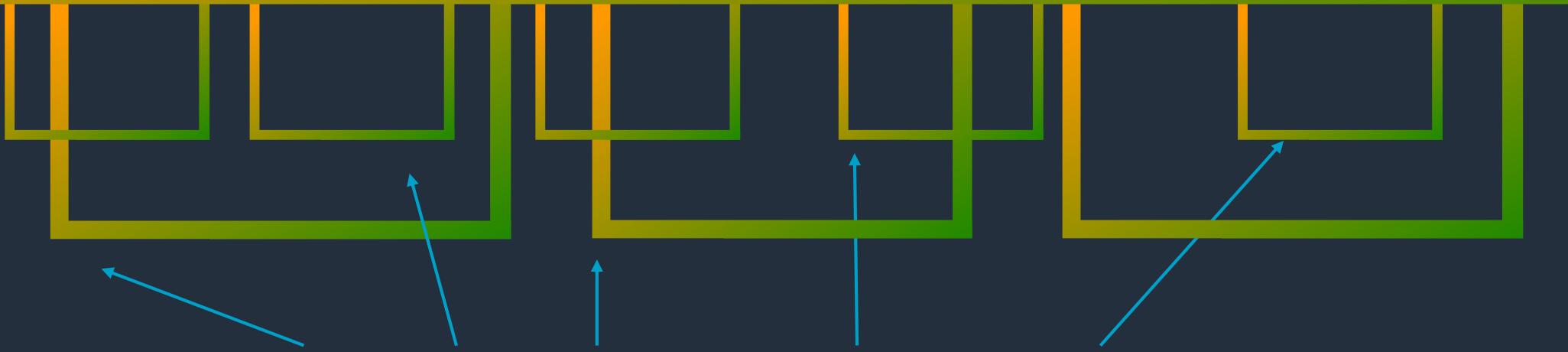
4dacd6d..4fa0318 master -> master



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Branching 전략

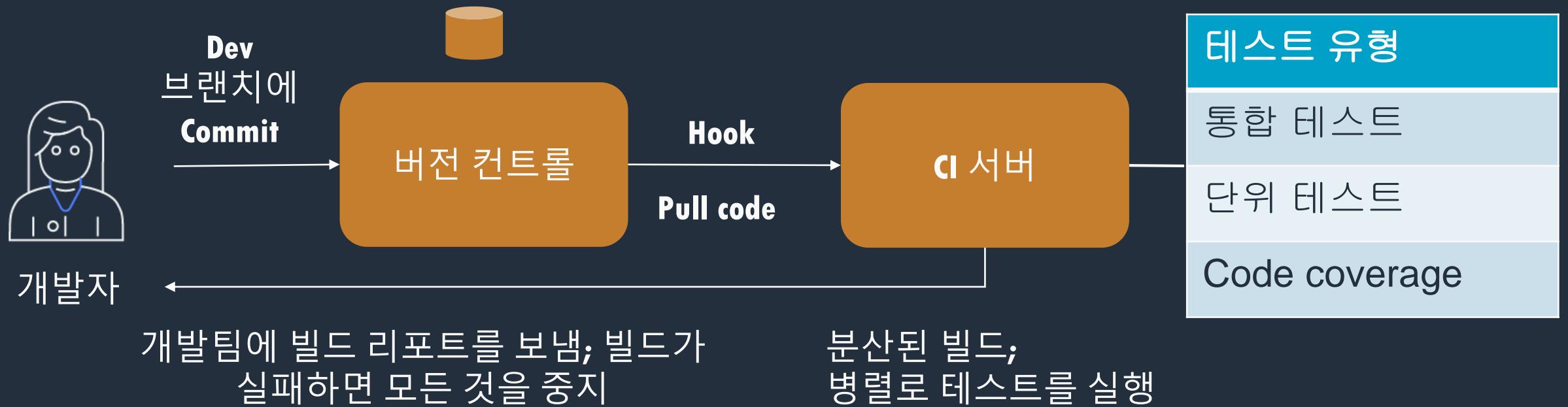
전체 개발 팀은 **Trunk (or Master)** 라는 브랜치를 공유



개발자들은 수명이 짧은 기능의 브랜치를 생성하고,
Pull 요청을 통해 병합 (Merge)

지속적 통합 (CI)

지속적 통합 workflow



AWS CodeBuild

지속적인 확장으로 코드 작성 및 테스트



- 소스 코드를 컴파일하고, 테스트를 실행하며, 소프트웨어 패키지를 생성하는 완전 관리형 빌드 서비스
- 지속적으로 확장하고 여러 빌드를 동시에 처리
- **Docker** 이미지를 통해 필요에 맞는 맞춤형(**custom**) 빌드 환경을 제공 할 수 있음
- 사용하는 컴퓨팅 리소스에 대해 분 단위로만 지불
- **AWS CodePipeline** 및 **Jenkins** 통합으로 시작

AWS CodeBuild 동작 방법

소스 코드를 컴파일하고, 테스트를 실행하며, 배포 준비가 된 소프트웨어 패키지를 생성하는 완전 관리형 빌드 서비스



소스 코드
다운로드



빌드 명령어
실행

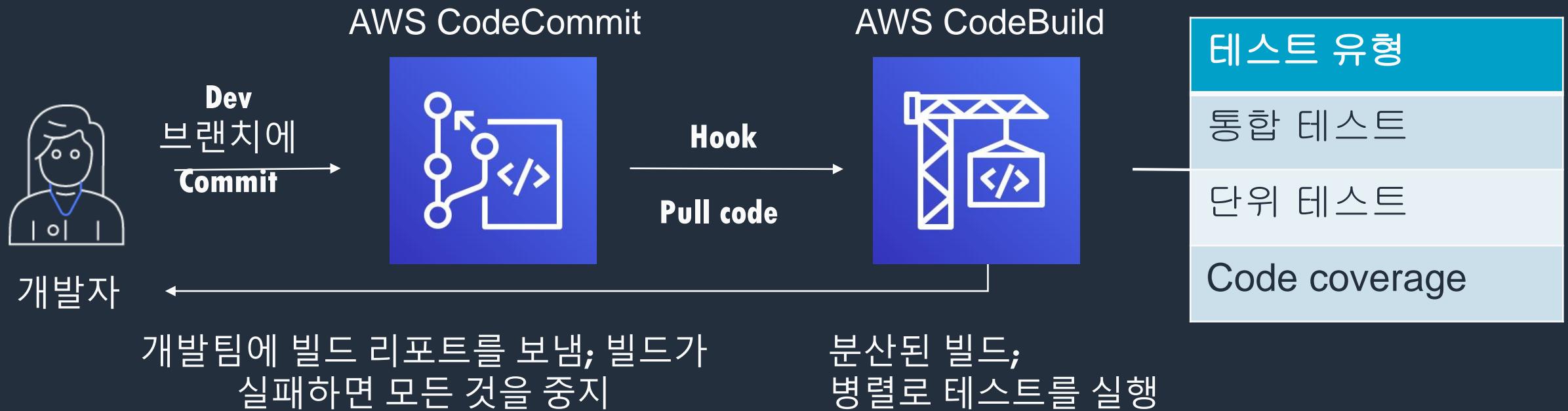


빌드 진행상황
모니터링



S3로 아티팩트
업로드

지속적 통합 workflow on AWS



buildspec.yml 예시

```
version: 0.1

{
    environment_variables:
        plaintext:
            JAVA_HOME: "/usr/lib/jvm/java-8-openjdk-amd64"

    phases:
        install:
            commands:
                - apt-get update -y
                - apt-get install -y maven
        pre_build:
            commands:
                - echo Nothing to do in the pre_build phase...
        build:
            commands:
                - echo Build started on `date`
                - mvn install
        post_build:
            commands:
                - echo Build completed on `date`
    artifacts:
        type: zip
        files:
            - target/messageUtil-1.0.jar
    discard-paths: yes
}
```

- } 빌드 단계에 사용될 변수
- } 빌드 단계를 명시
- **Dependencies 설치**
 - **Pre-build/Post-build 단계**
- } 아티팩트를 생성하고 S3에 저장



빌드를 요구하는 것

코드의 “**Building**”은 일반적으로 컴파일 된 바이너리가 필요한 언어를 나타냄:

- **.NET languages: C#, F#, VB.net, etc.**
- **Java and JVM languages: Java, Scala, JRuby**
- **Go**
- **iOS languages: Swift, Objective-C**

또한 **Docker** 컨테이너 이미지를 만드는 과정을 **Image Building**이라고 함

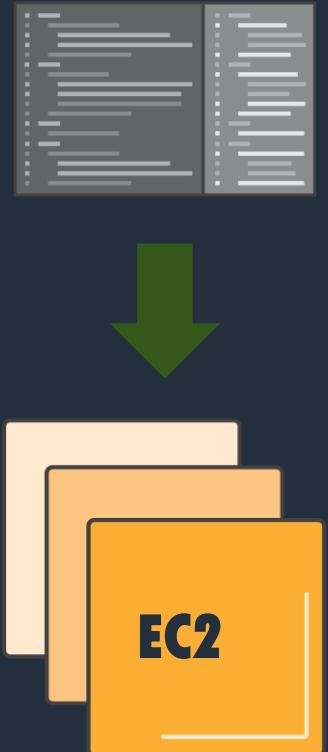


빌드를 요구하지 않는 것

많은 언어가 빌드를 요구하지 않음. 이들은 해석 된 언어 (**Interpreted**)로 간주됨:

- **PHP**
- **Ruby**
- **Python**
- **Node.js**

단지 코드를 배포하면 됨!



코드 테스트

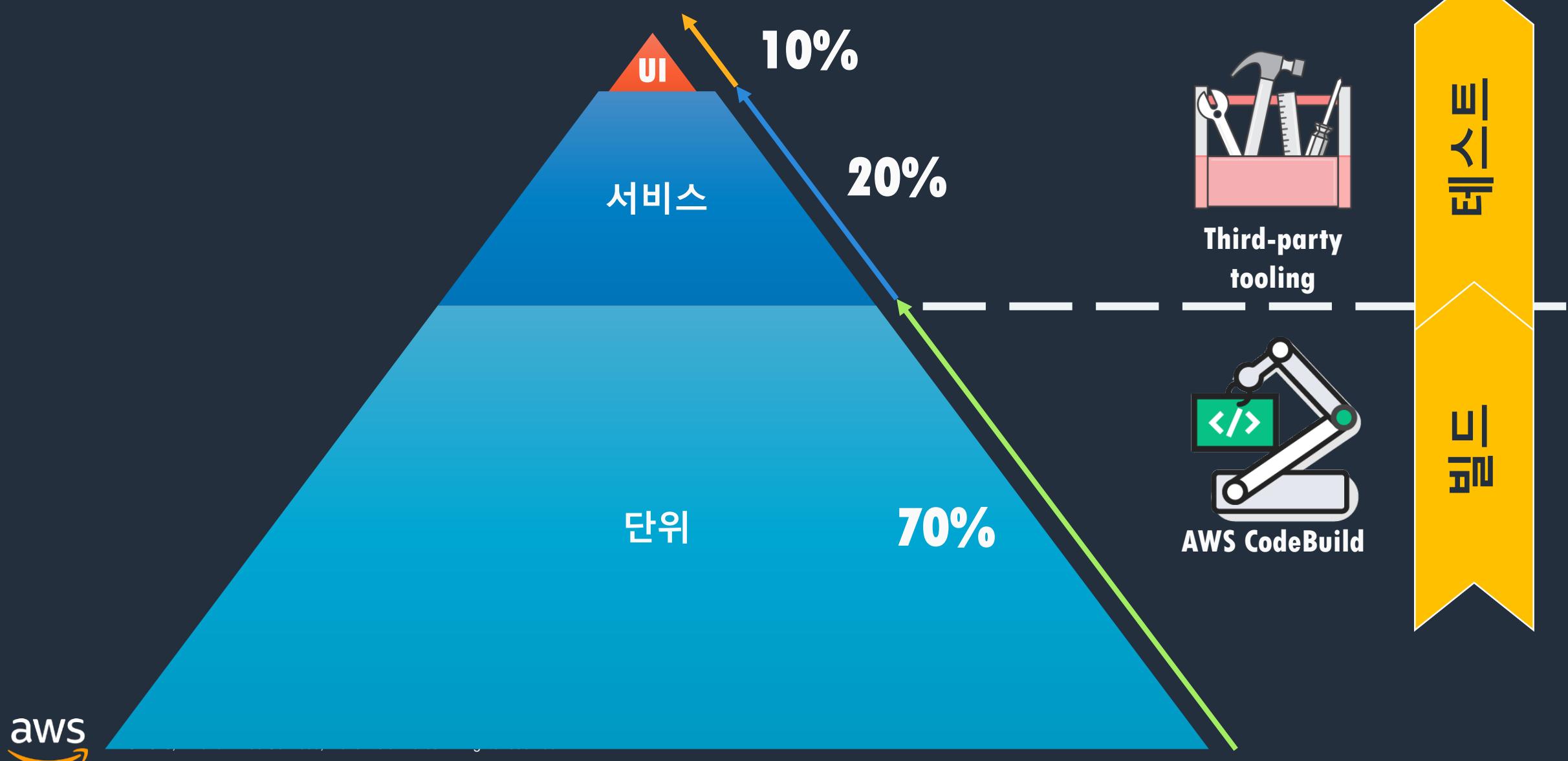
테스트는 과학과 예술의 형태!

코드 테스트 목표:

- 의도한 기능을 확인하고 싶음
- 프로그래밍 구문 오류 포착
- 코드 패턴 및 형식 표준화
- 원치 않는 어플리케이션 사용률 및 로직 에러로 인한 **Bug** 감소
- 보다 안전한 어플리케이션 만들기



서비스 및 릴리스 단계 별 해당하는 테스트

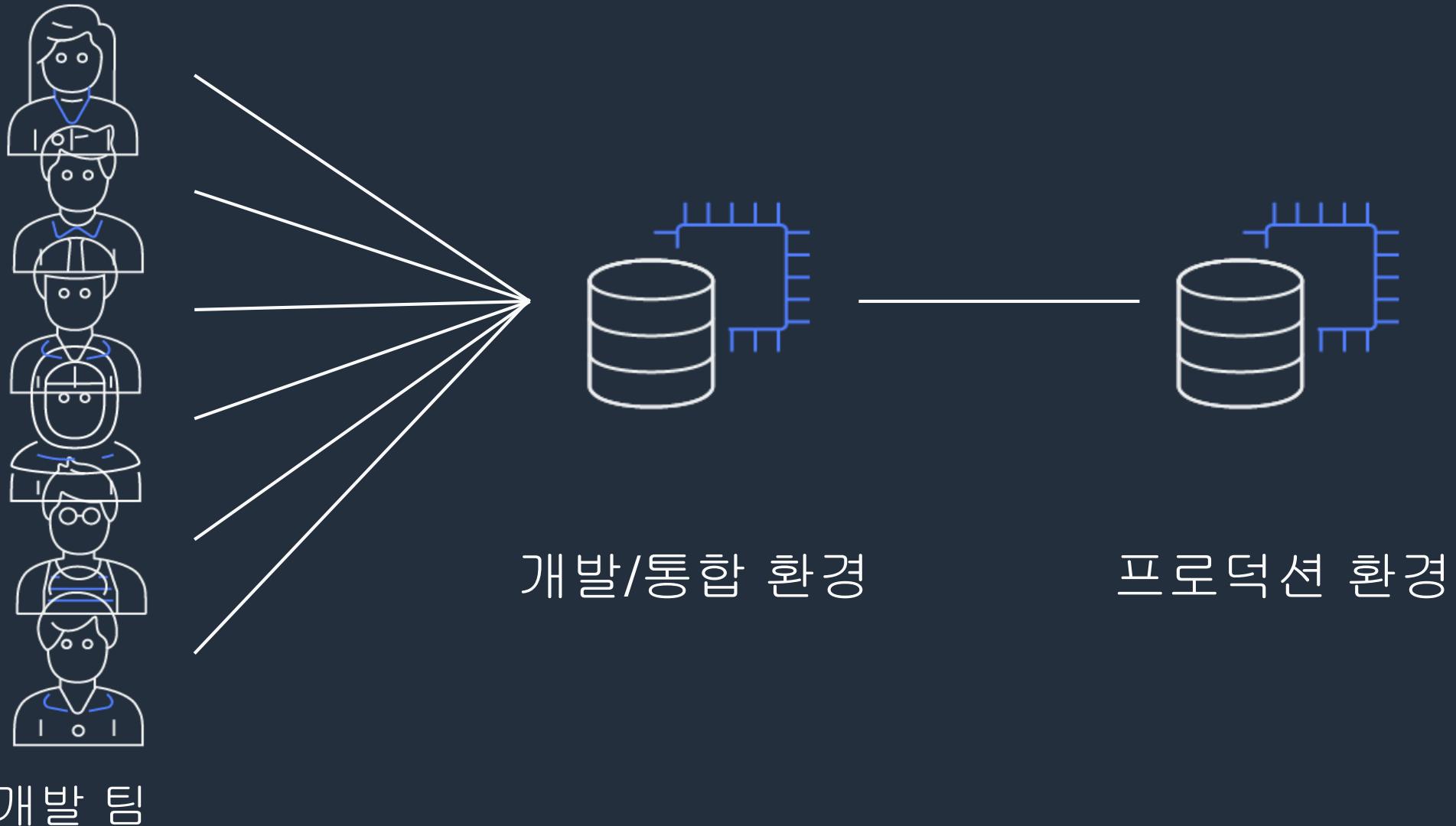


코드로 써의 인프라 (IaC)



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

하나의 개발 환경은 확장되지 않음



개발자 당 하나의 개발 / 통합 환경



개발 팀

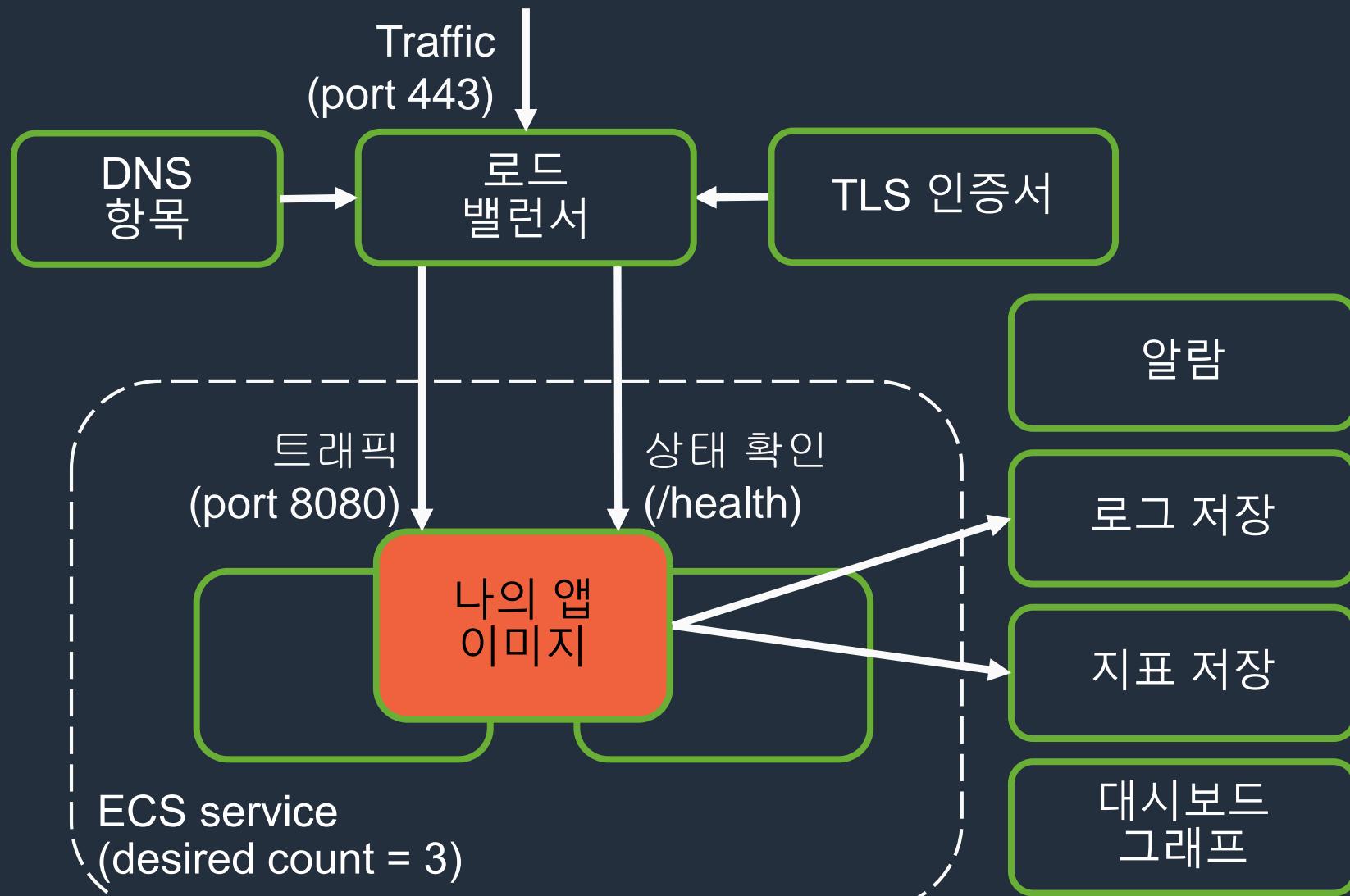


개발/통합 환경

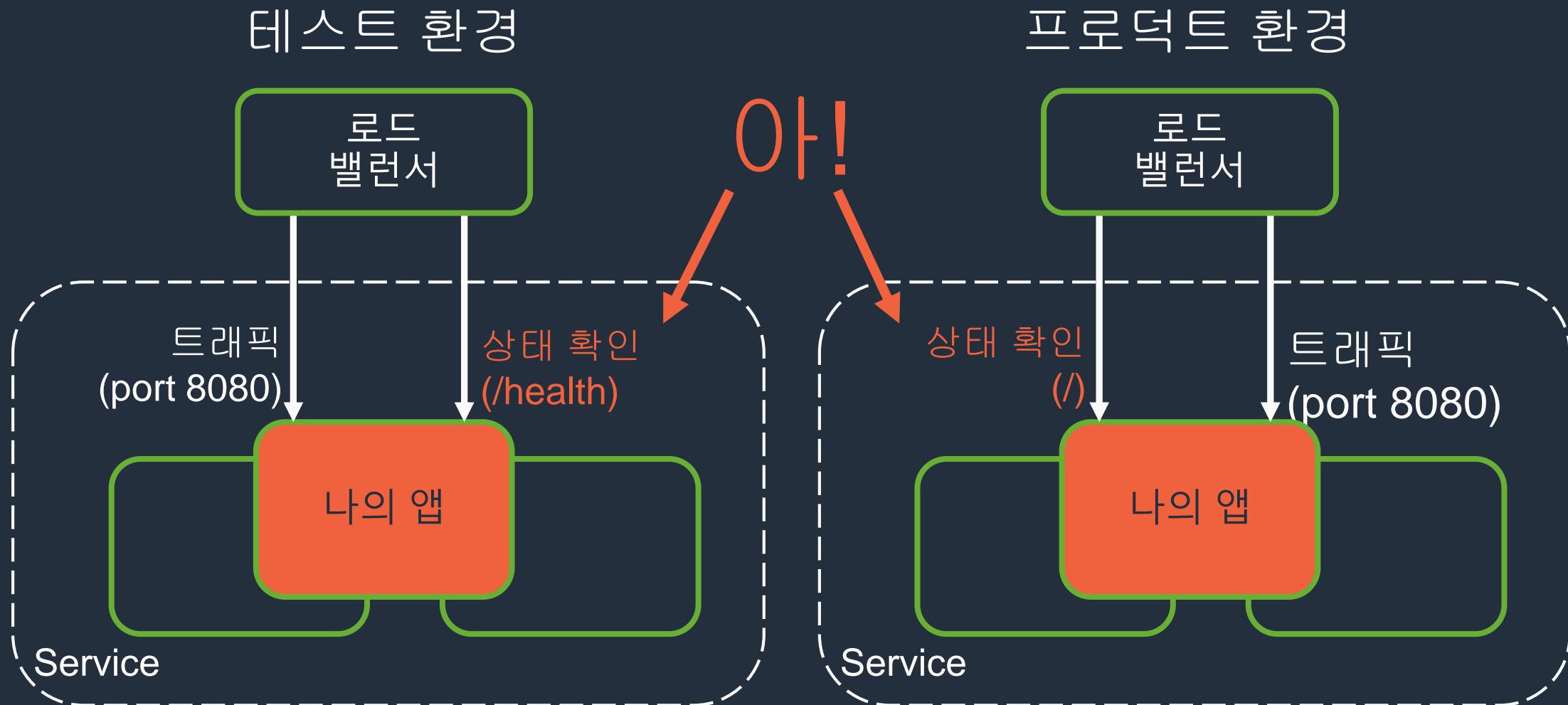


프로덕션 환경

어플리케이션에 영향을 주는 변수



어플리케이션 릴리스 자동화의 한계



AWS CloudFormation



AWS CloudFormation

- 코드로서의 인프라 (**IaC**)
- 버전 관리와 통합
- JSON, YAML 형식
- 템플릿
- 스택
- 광범위한 AWS 리소스 지원
- AWS CloudFormation Designer – 비주얼 도구

AWS CloudFormation: 어플리케이션 스택 예시

템플릿을 사용하여
개발, 테스트 및
프로덕션과 같은 여러
환경을 구축



선택한 버전 관리
시스템을 사용하여 이
템플릿의 변경 사항을
저장하고 추적.

전체 애플리케이션은 AWS
CloudFormation 템플릿 내에
나타낼 수 있음

템플릿 구조

```
{  
  "Description" : "Create an EC2 instance.",  
  "Resources" : {  
    "Ec2Instance" : {  
      "Type" : "AWS::EC2::Instance",  
      "Properties" : {  
        "KeyName" : "my-key-pair",  
        "ImageId" : "ami-75g0061f",  
        "InstanceType" : "m1.medium"  
      }  
    }  
  }  
}
```

템플릿 구조

```
{  
  "Description" : "Create an EC2 instance.",  
  "Parameters" : {  
    "UserKeyName" : {  
      "Description" : "The EC2 Key Pair to allow SSH access to the instance",  
      "Type" : "String"  
    }  
  },  
  "Resources" : {  
    "Ec2Instance" : {  
      "Type" : "AWS::EC2::Instance",  
      "Properties" : {  
        "KeyName" : { "Ref" : "UserKeyName"},  
        "ImageId" : "ami-75g0061f",  
        "InstanceType" : "m1.medium"  
      }  
    }  
  }  
}
```



템플릿 구조

```
{  
  "Description" : "Create an EC2 instance.",  
  "Parameters" : {  
    "UserKeyName" : {  
      "Description" : "The EC2 Key Pair to allow SSH access to the instance",  
      "Type" : "String"  
    },  
    "InstanceType" : {  
      "Description" : "The EC2 Instance Type to launch.",  
      "Type" : "String",  
      "AllowedValues" : [ "t1.micro", "m1.small", "m1.medium" ]  
    }  
  },  
  "Resources" : {  
    "Ec2Instance" : {  
      "Type" : "AWS::EC2::Instance",  
      "Properties" : {  
        "KeyName" : { "Ref" : "UserKeyName" },  
        "ImageId" : "ami-75g0061f",  
        "InstanceType" : { "Ref" : "InstanceType" }  
      }  
    }  
  },  
  "Outputs" : {  
    "InstancePublicDnsName" : {  
      "Description" : "The public DNS name of the newly created EC2 instance",  
      "Value" : { "Fn::GetAtt" : [ "Ec2Instance", "PublicDnsName" ] }  
    }  
  }  
}
```



어플리케이션 배포 - User Data

```
"UserData": {  
    "Fn::Base64": {  
        "Fn::Join": [  
            "",  
            [  
                "#!/bin/bash -ex\n",  
                "yum -y install git-core\n",  
                "yum -y install php-pear\n",  
                "pear install Crypt_HMAC2-1.0.0\n",  
                "pear install HTTP_Request-1.4.4\n",  
                "pear install aws/sdk\n",  
            ]  
        ]  
    }  
}
```



어플리케이션 배포 - cfn-init

```
"Ec2Instance": {  
    "Metadata": {  
        "AWS::CloudFormation::Init": {  
            "config": {  
                "sources" : {  
                    "/usr/local/bin/s3cmd" : "https://github.com/s3tools/s3cmd"  
                },  
                "packages": {  
                    "yum": { "git": [] }  
                }  
            }  
        }  
    }  
}
```



AWS CloudFormation Designer – 시각화 도구

AWS Services Edit Pubali Sen N. Virginia Support

File: 'new.template'

Resource types

- DynamoDB
- EC2
 - CustomerGateway
 - DHCPOptions
 - EIP
 - Instance
 - InternetGateway
 - NetworkAcl

Properties Metadata DeletionPolicy DependsOn Condition Errors

IGW2O4KV

```
1 {
2   "Resources": {
3     "IGW2O4KV": {
4       "Type": "AWS::EC2::InternetGateway",
5       "Properties": {}
6     }
7   }
}
```

Components Template

The screenshot shows the AWS CloudFormation Designer interface. On the left, there's a sidebar titled 'Resource types' with a tree view of available services: DynamoDB and EC2. Under EC2, several sub-resources are listed: CustomerGateway, DHCPOptions, EIP, Instance, InternetGateway, and NetworkAcl. The main workspace is titled 'File: 'new.template'' and contains a visual representation of a network stack. It includes a central orange square resource, a white square resource with a cloud icon, and another white square resource with a hexagonal icon. These resources are interconnected by dashed lines with blue circular endpoints. To the right of the workspace is a vertical toolbar with a search icon, a plus sign for adding resources, and a refresh icon. Below the workspace is a tab bar with 'Properties', 'Metadata', 'DeletionPolicy', 'DependsOn', 'Condition', and 'Errors'. The 'Properties' tab is currently selected. A large code editor window below shows the JSON template for a new Internet Gateway resource named 'IGW2O4KV'. The code is as follows:

```
1 {
2   "Resources": {
3     "IGW2O4KV": {
4       "Type": "AWS::EC2::InternetGateway",
5       "Properties": {}
6     }
7   }
}
```

At the bottom of the interface, there are tabs for 'Components' and 'Template', with 'Template' being the active tab.

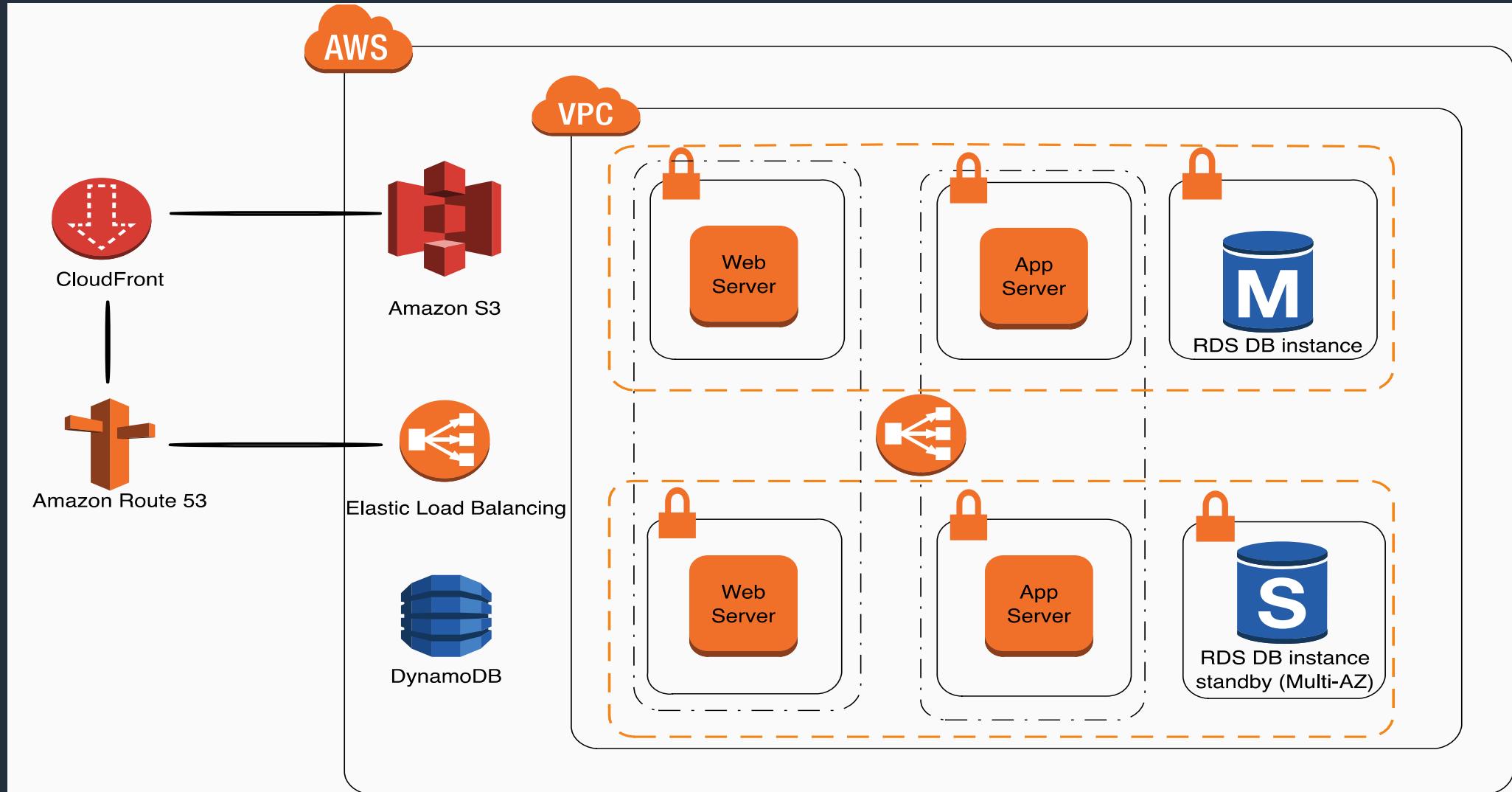


3rd Party 도구

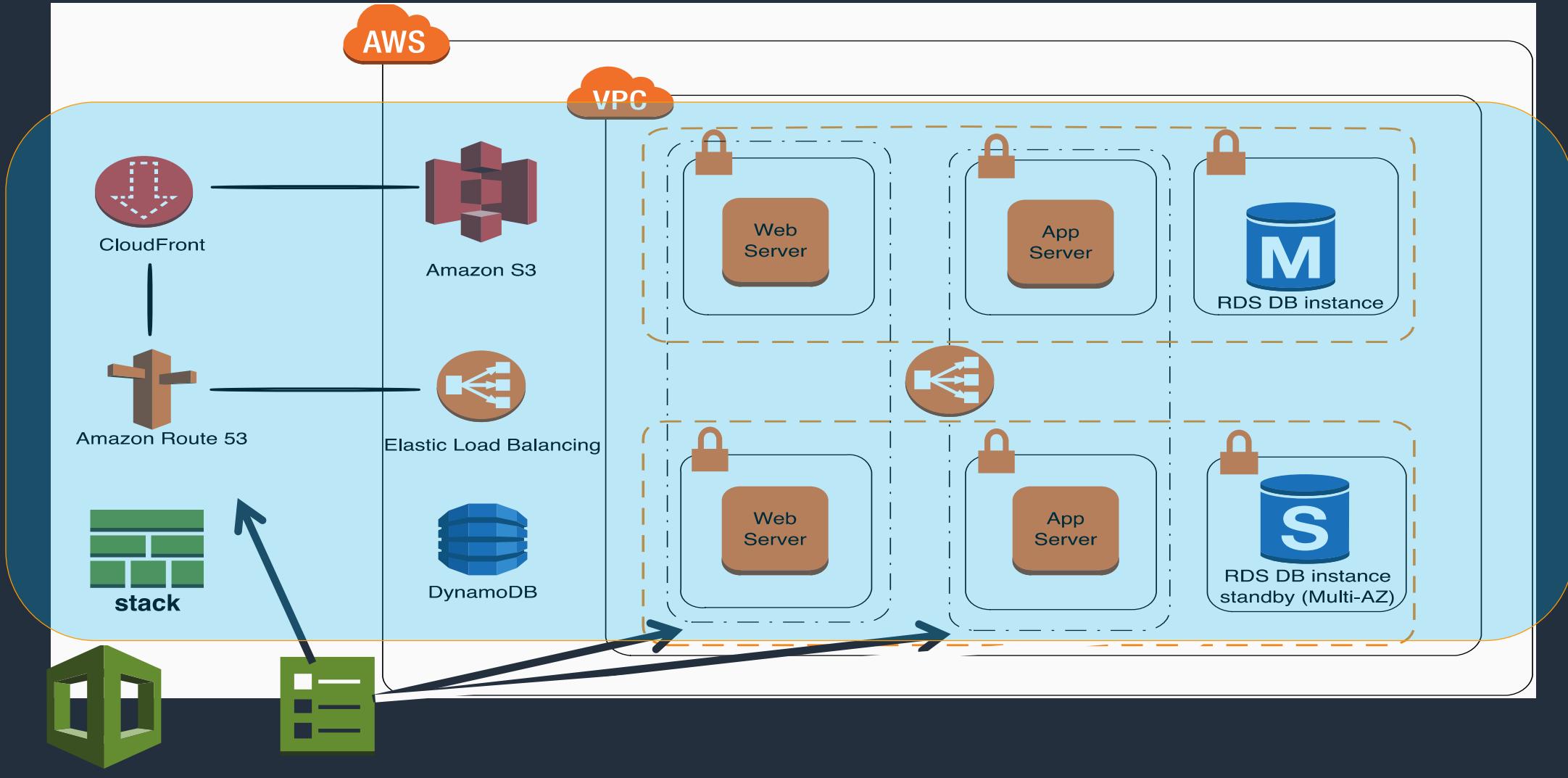
- 기존 구성 관리 도구 (Configuration Management Tools) 와 쉽게 통합
- User-Data 또는 cfn-init를 사용하여 쉽게 에이전트를 구성



샘플 아키텍처



CloudFormation을 사용한 샘플 아키텍처

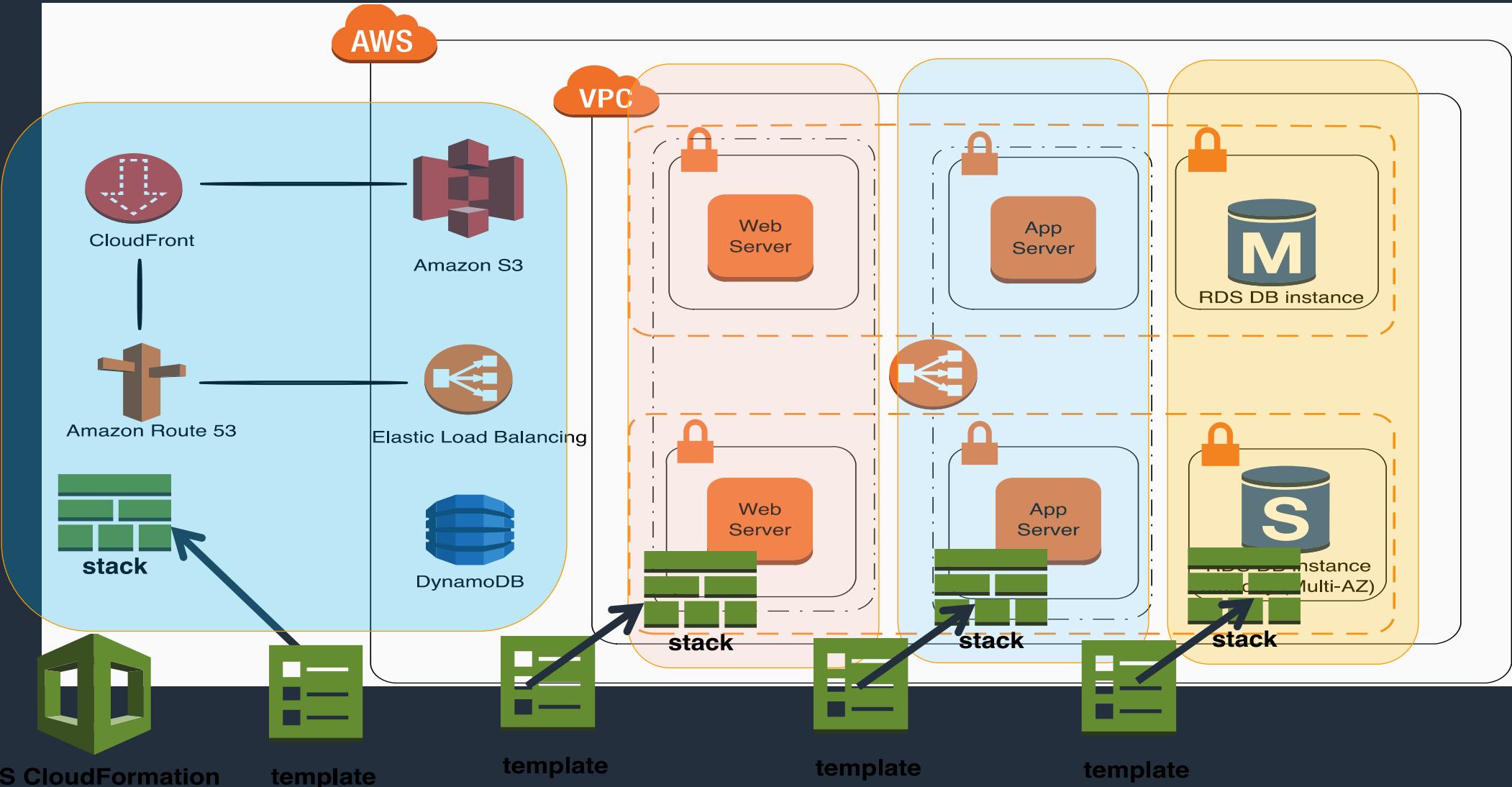


AWS CloudFormation

template



CloudFormation을 사용한 샘플 아키텍처



AWS Cloud Development Kit (AWS CDK)

재사용 가능한 구성 요소로 클라우드 인프라를 모델링하기 위한 *multi-language* 소프트웨어 개발 프레임 워크



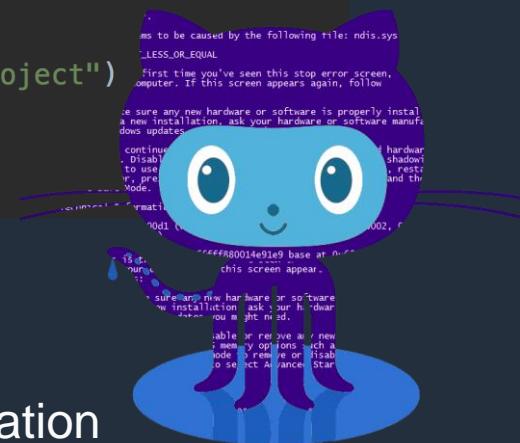
```
#!/usr/bin/env node
import ...

export class MyStack extends cdk.Stack {
  constructor(scope: cdk.App, id: string, props?: cdk.StackProps) {
    super(scope, id, props);

    let myVpc = new ec2.Vpc(this, "Vpc", {
      maxAZs: 2,
    });

    let myCluster = new ecs.Cluster(this, 'Cluster', {
      vpc: myVpc,
    });

    // Define an ECS Fargate Service that runs two instances
    // of a Docker image created from a local Dockerfile
    new ecs_patterns.LoadBalancedFargateService(this, 'Service', {
      cluster: myCluster,
      desiredCount: 2,
      image: ecs.AssetImage.fromAsset(directory: "my-docker-project")
    });
  }
}
```



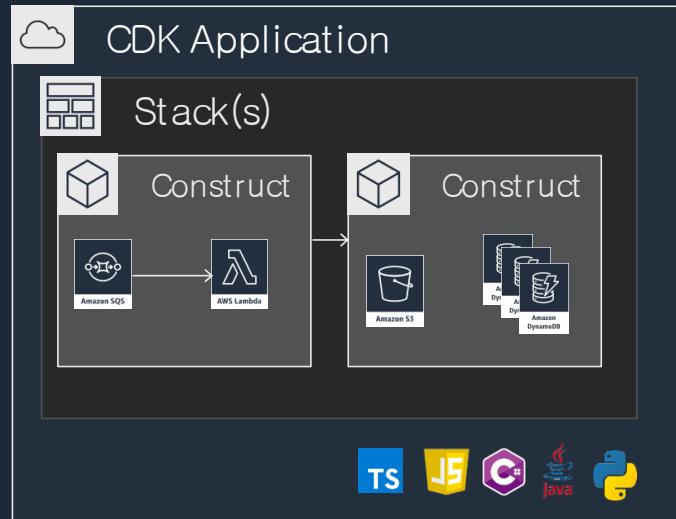
AWS CloudFormation



AWS Cloud Development Kit (AWS CDK)

주요 구성 요소

Core Framework



AWS Construct Library



AWS CDK CLI

A screenshot of a terminal window showing the AWS CDK CLI in action. The command `cdk init --language typescript sample-app` is run, initializing a new project template. The terminal also displays npm installation logs and a list of useful commands:

```
~/Projects/hello-cdk
> cdk init --language typescript sample-app
Applying project template sample-app for typescript
Initializing a new git repository...
Executing npm install...
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN hello-cdk@0.1.0 No repository field.
npm WARN hello-cdk@0.1.0 No license field.

# Useful commands
* `npm run build`      compile typescript to js
* `npm run watch`       watch for changes and compile
* `cdk deploy`          deploy this stack to your default AWS account/region
* `cdk diff`            compare deployed stack with current state
* `cdk synth`           emits the synthesized CloudFormation template

~/Projects/hello-cdk master* 33s
>
```



CDK: VPC 생성

```
//  
// create VPC w/ public and private subnets in 2 AZ  
// this also creates a NAT Gateway  
  
//  
const vpc = new ec2.Vpc(this, 'NewsBlogVPC', {  
    maxAzs : 2  
} );
```



CDK: Application 패키징

```
//  
// create static web site as S3 assets  
  
var path = require('path');  
  
const asset = new assets.Asset(this, 'YourSampleApp', {  
    path: path.join(__dirname, '../html')  
});
```



CDK: 서버 Bootstrapping

```
// define a user data script to install & launch our app
const userData = UserData.forLinux();

userData.addCommands('yum install -y nginx',
    'chkconfig nginx on', 'service nginx start');

userData.addCommands(`aws s3 cp s3://${asset.s3BucketName}/${asset.s3ObjectKey} .`,
    `unzip *.zip`,
    `/bin/cp -r -n ${env}/* /usr/share/nginx/html/`);
```

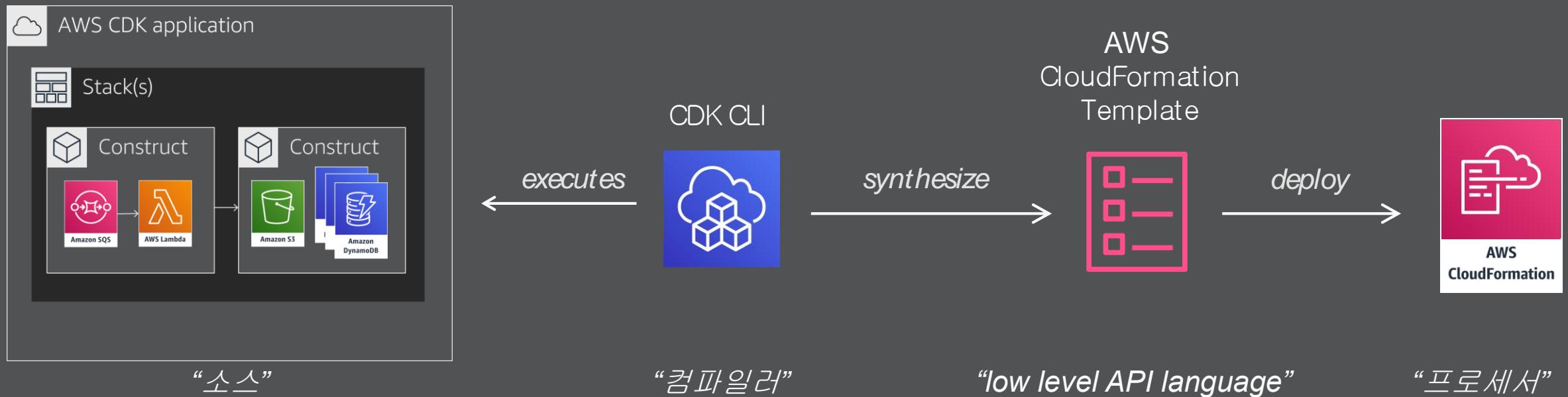


CDK: Auto Scaling Group 생성

```
// create an auto scaling group for each environment
const asg = new autoscaling.AutoScalingGroup(this, 'YourAppAutoScalingGroup', {
    vpc,
    instanceType: ec2.instanceType.of(ec2.instanceClass.BURSTABLE3,
        ec2.instanceSize.MICRO),
    machineImage: new ec2.AmazonLinuxImage(),
    desiredCapacity: 2,
    role: role,
    userData: userData
}) ;
```

CDK: 개발(dev) 환경 배포

\$ cdk deploy



Deployment



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS CodeDeploy

자동화 된 배포 조정

Application
revisions



v1, v2, v3



배포 그룹



- 단일 인스턴스에서 수천으로 확장
- **Zero-downtime** 배포를 위한 다중 구성
- 배포 제어 및 모니터링 중앙 집중화
- 구성 가능한 배치 후크 (**hooks**)
- **EC2**, 온 프레미스 및 **Lambda**에 배포

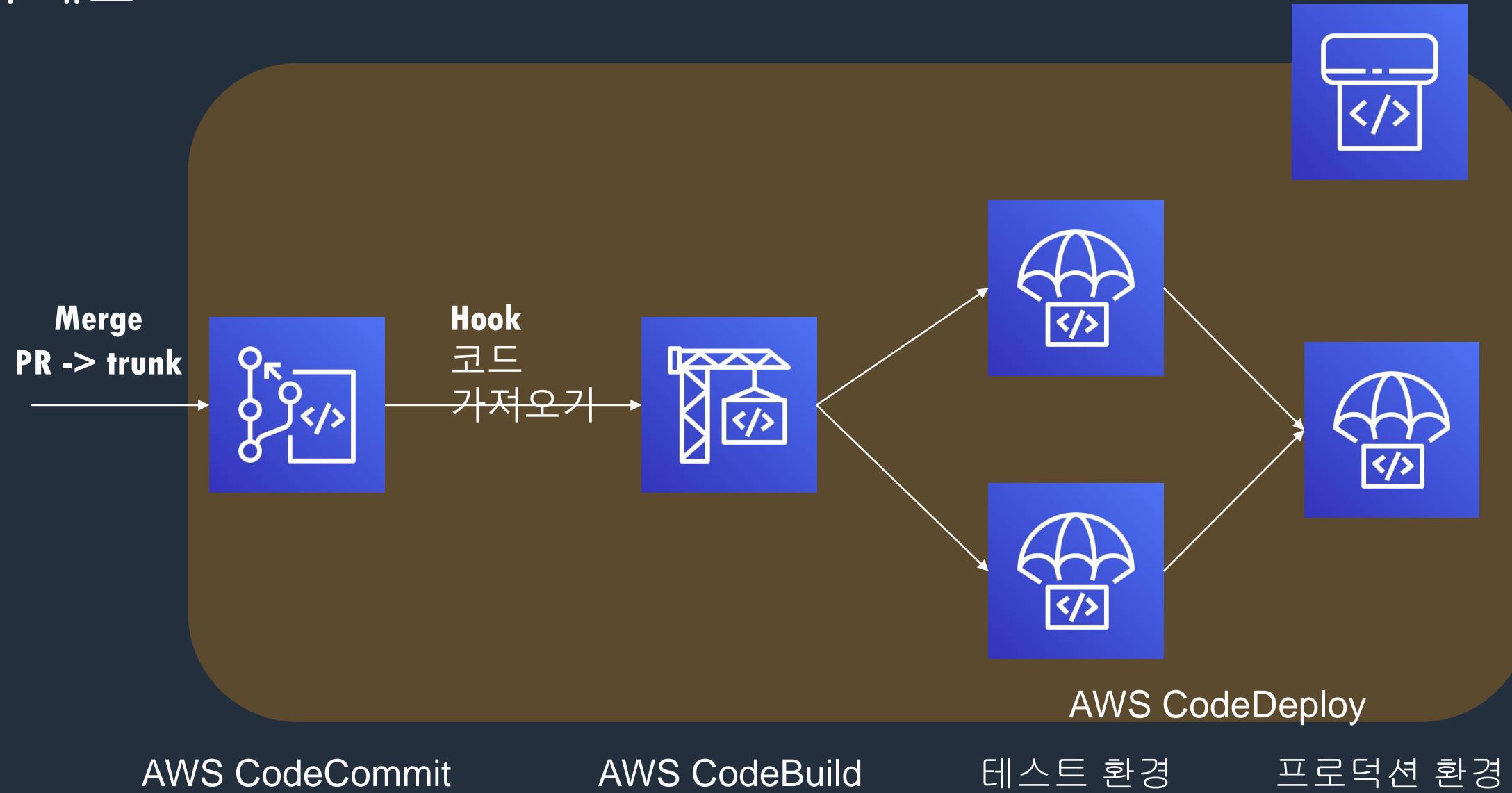
appspec.yml 예시

```
{  
    version: 0.0  
    os: linux  
    files:  
        - source: /  
          destination: /var/www/html  
    permissions:  
        - object: /var/www/html  
          pattern: “*.html”  
          owner: root  
          group: root  
          mode: 755  
    hooks:  
        ApplicationStop:  
            - location: scripts/deregister_from_elb.sh  
        BeforeInstall:  
            - location: scripts/install_dependencies.sh  
        ApplicationStart:  
            - location: scripts/start_httpd.sh  
        ValidateService:  
            - location: scripts/test_site.sh  
            - location: scripts/register_with_elb.sh
```

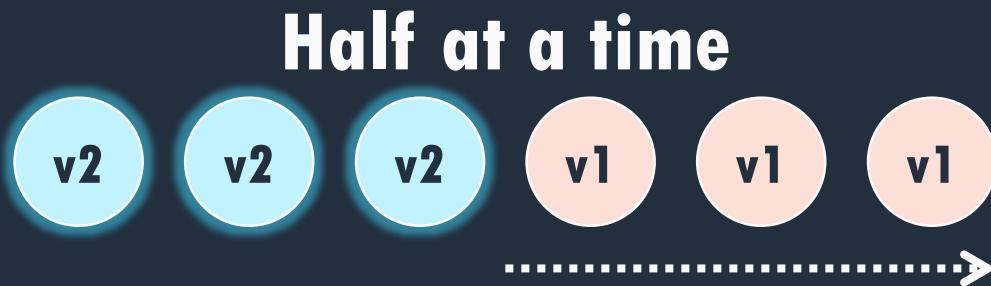
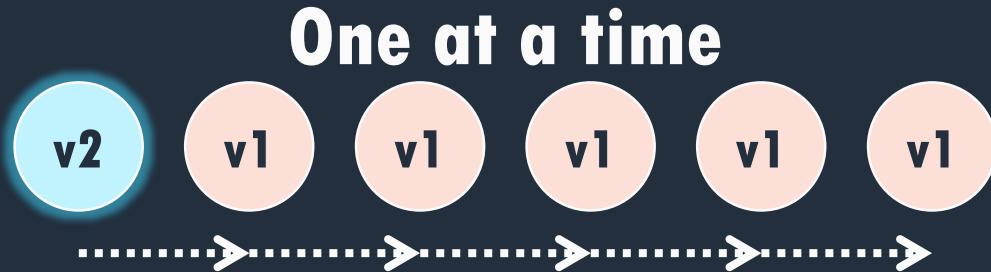
- 한 디렉토리로 응용 프로그램 파일을 보내고, 다른 디렉토리로 구성 파일을 보냄
- 특정 디렉토리 및 파일에 대한 특정 권한 설정
- ELB에 인스턴스 제거 / 추가
- Dependancy 패키지 설치
- Apache 시작
- 성공적인 배포를 확인
- 추가 작업!



지속적 배포



배포 속도와 그룹 선택

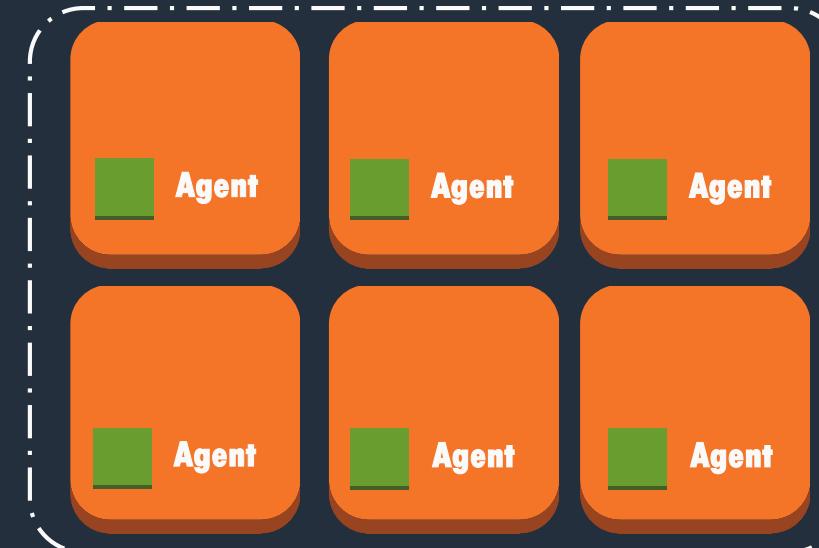


Dev 배포 그룹

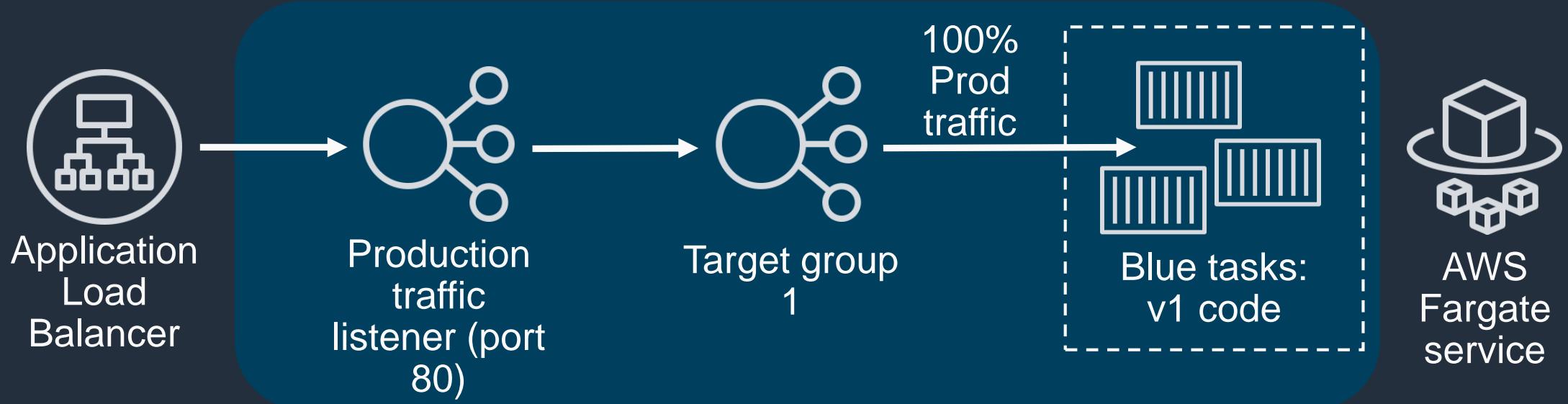


OR

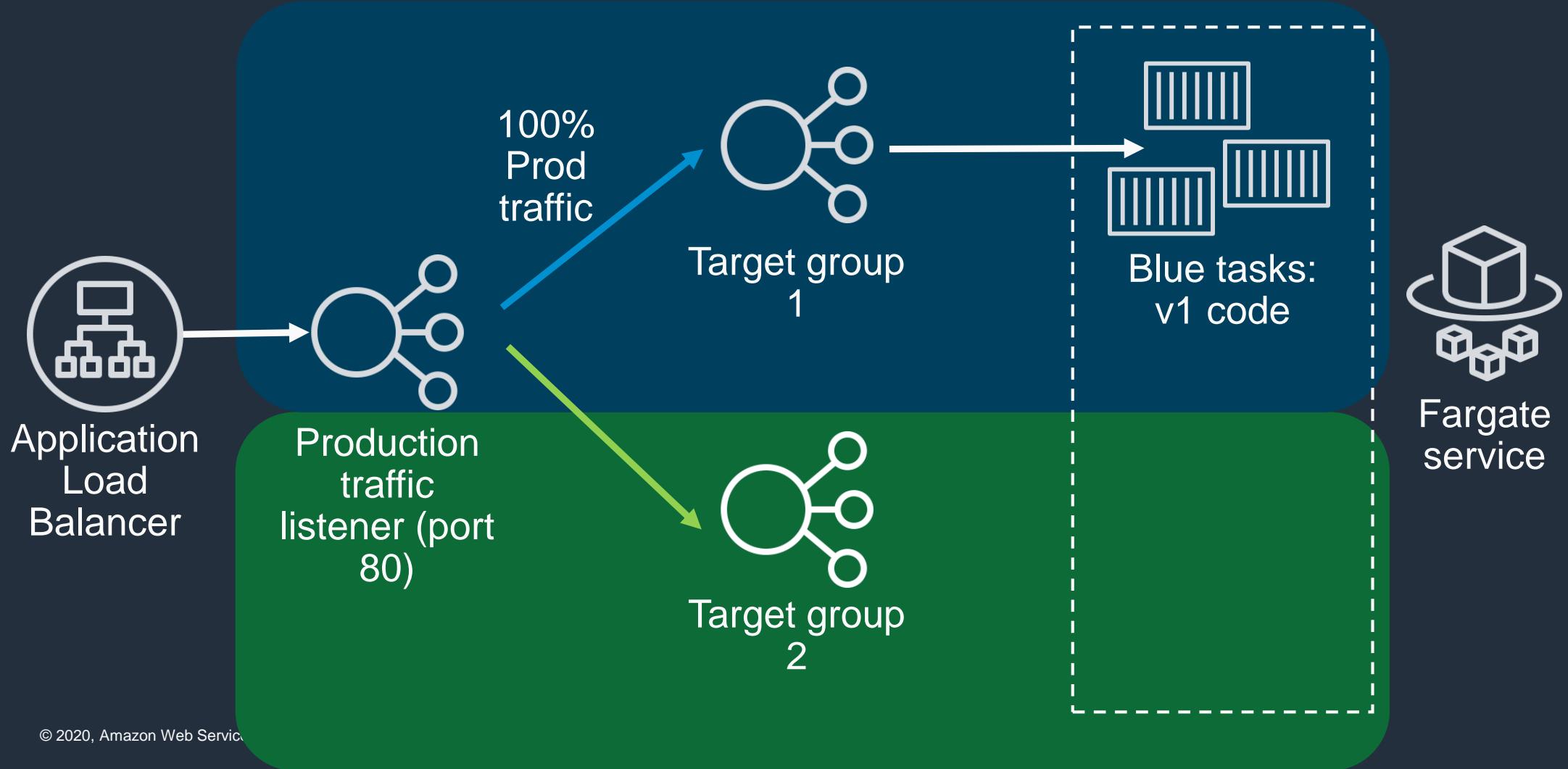
Prod 배포 그룹



블루-그린 배포

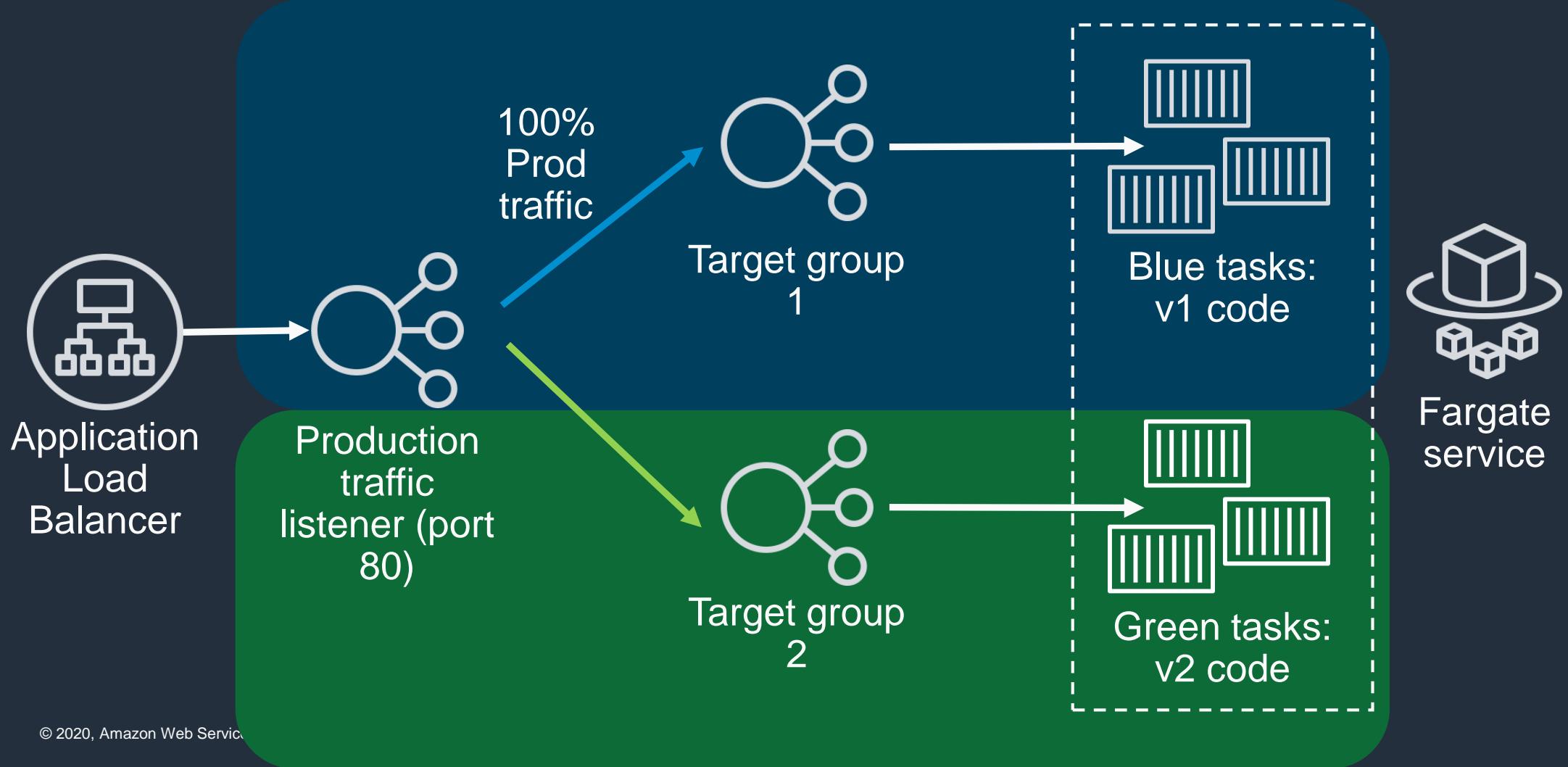


블루-그린 배포



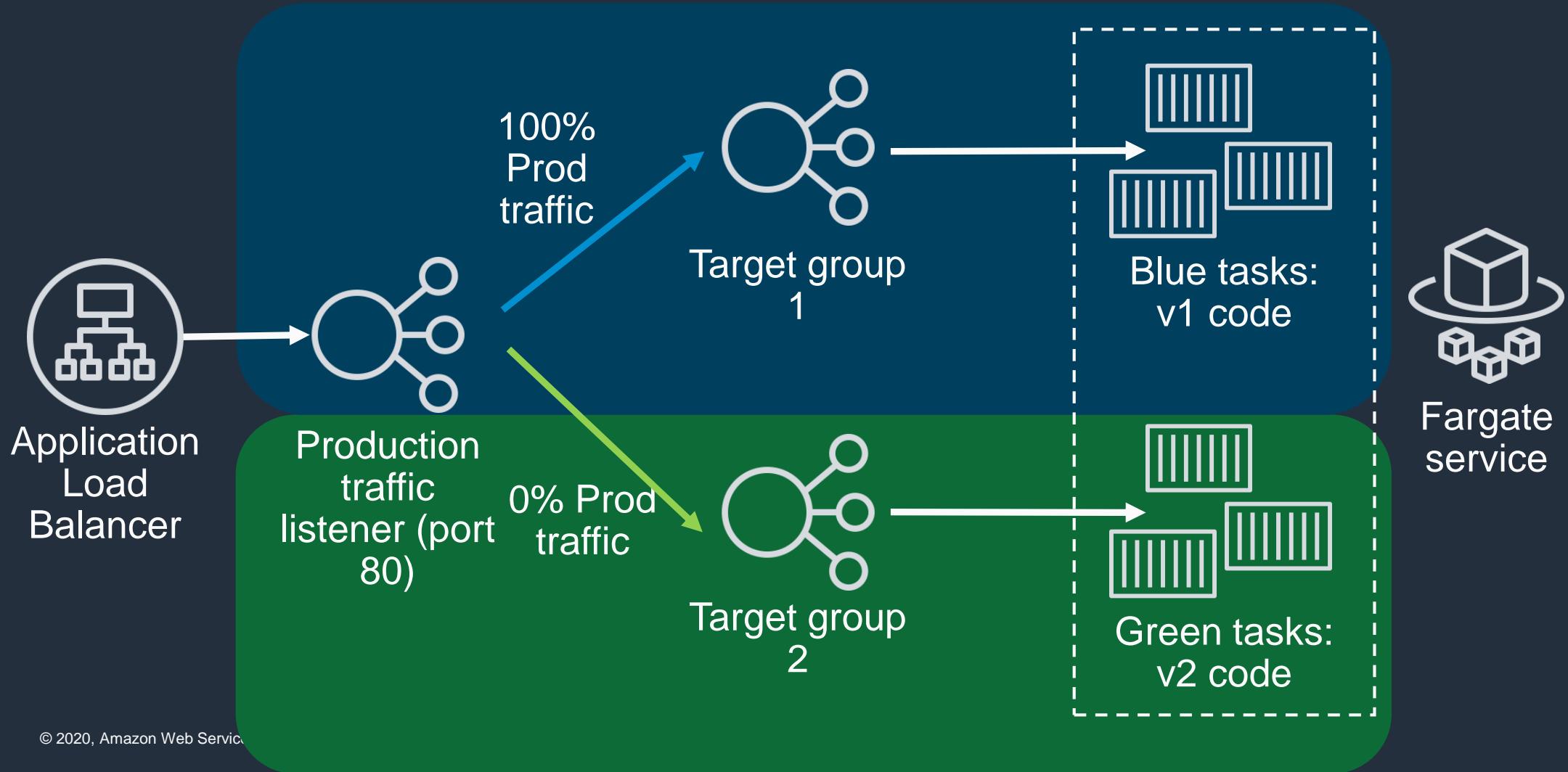
블루-그린 배포

Green Task 프로비전



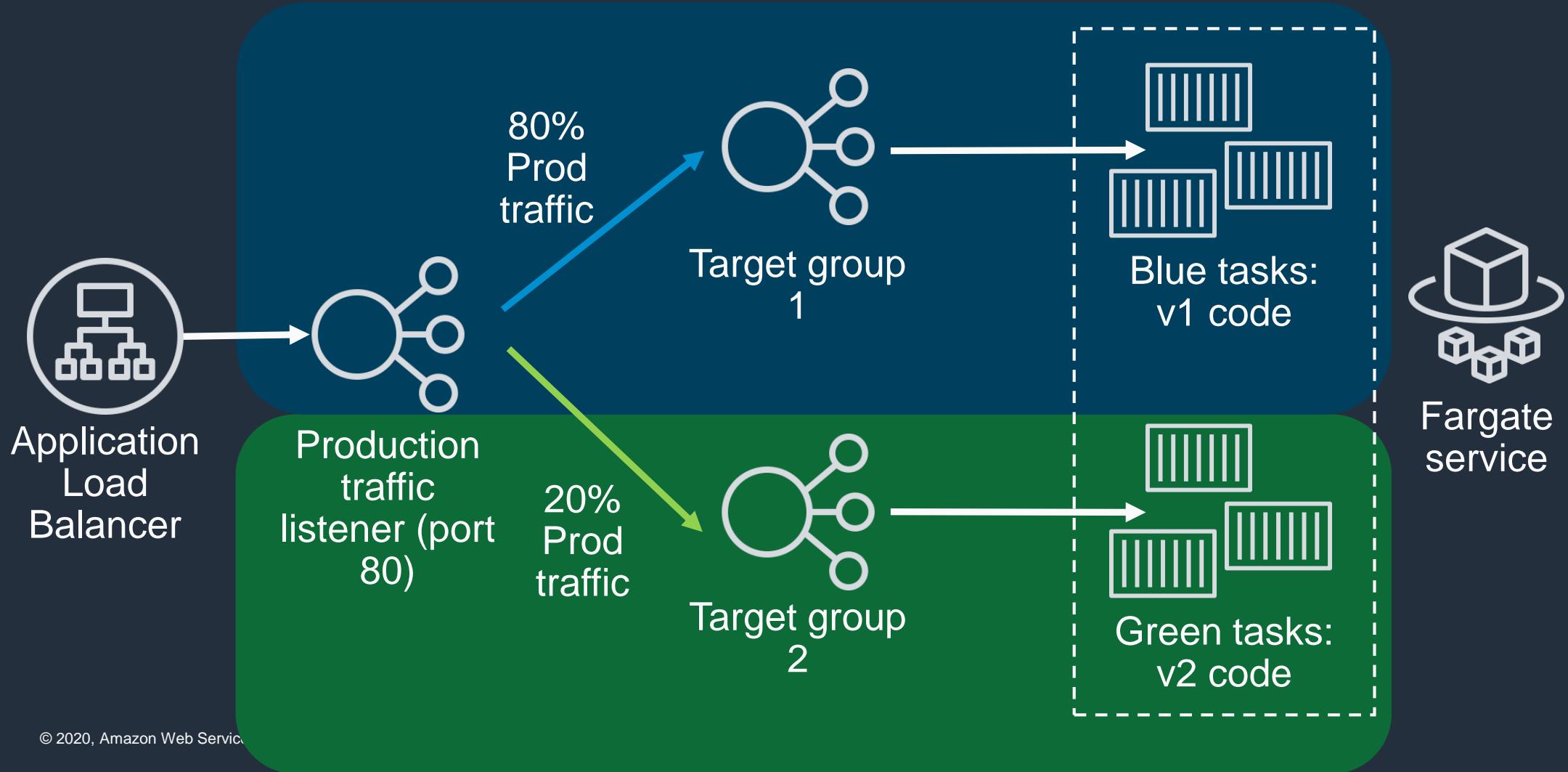
블루-그린 배포

Green Task가 프로덕션 트래픽을 받기 전에 테스트 엔드 포인트에 Hook 실행



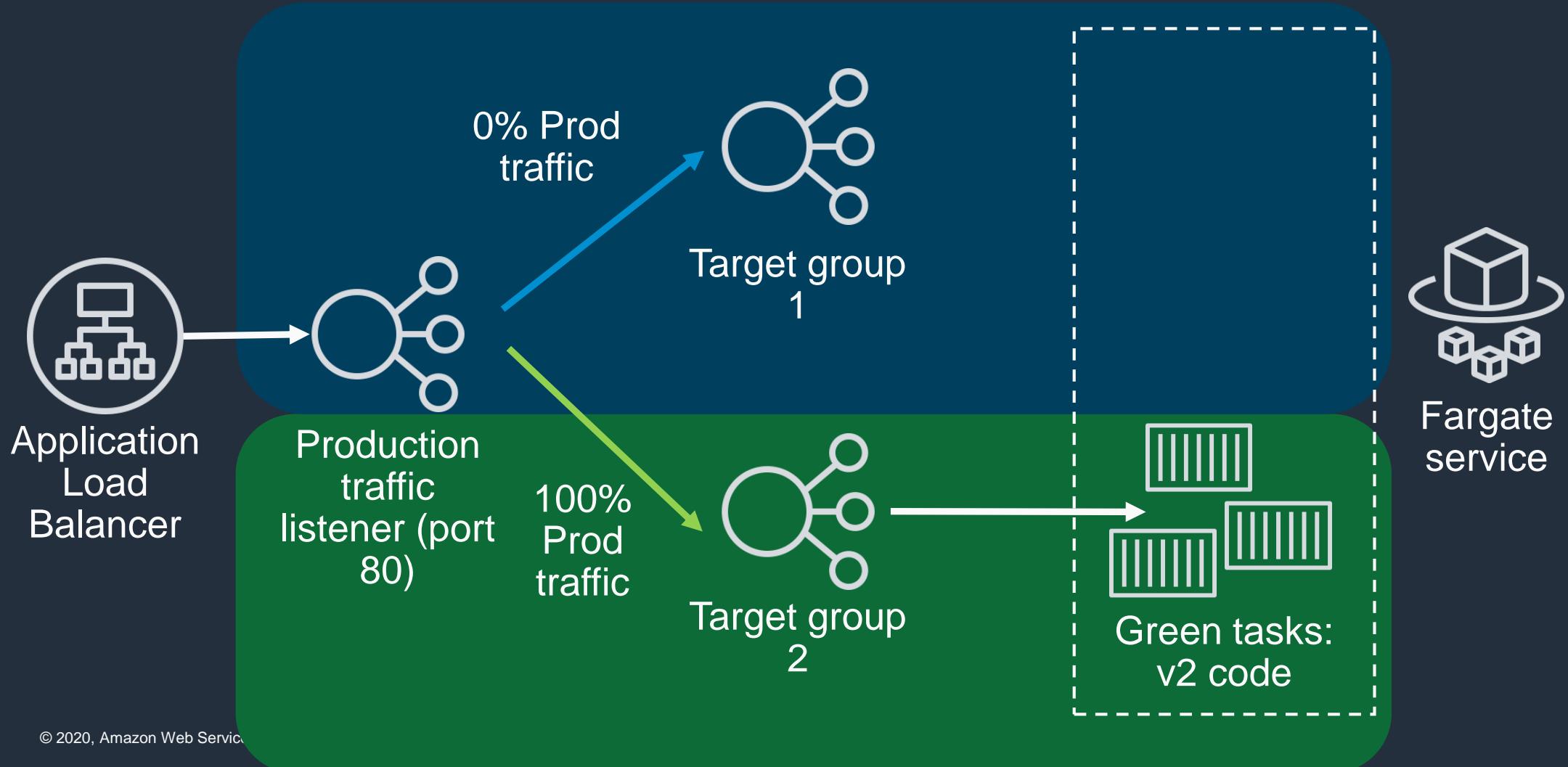
블루-그린 배포

경보(alarm)가 발생하면, Green Task 의 트래픽을 전환하고 롤백



블루-그린 배포

Drain blue tasks

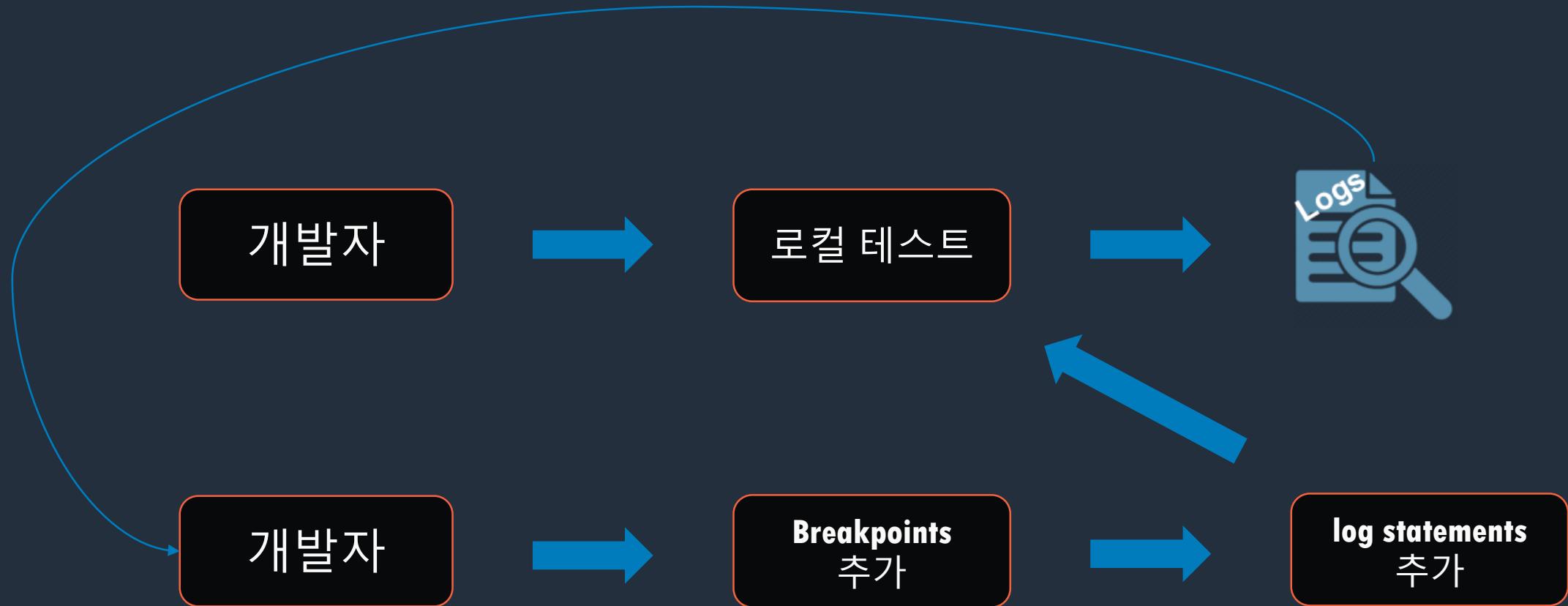


Debugging

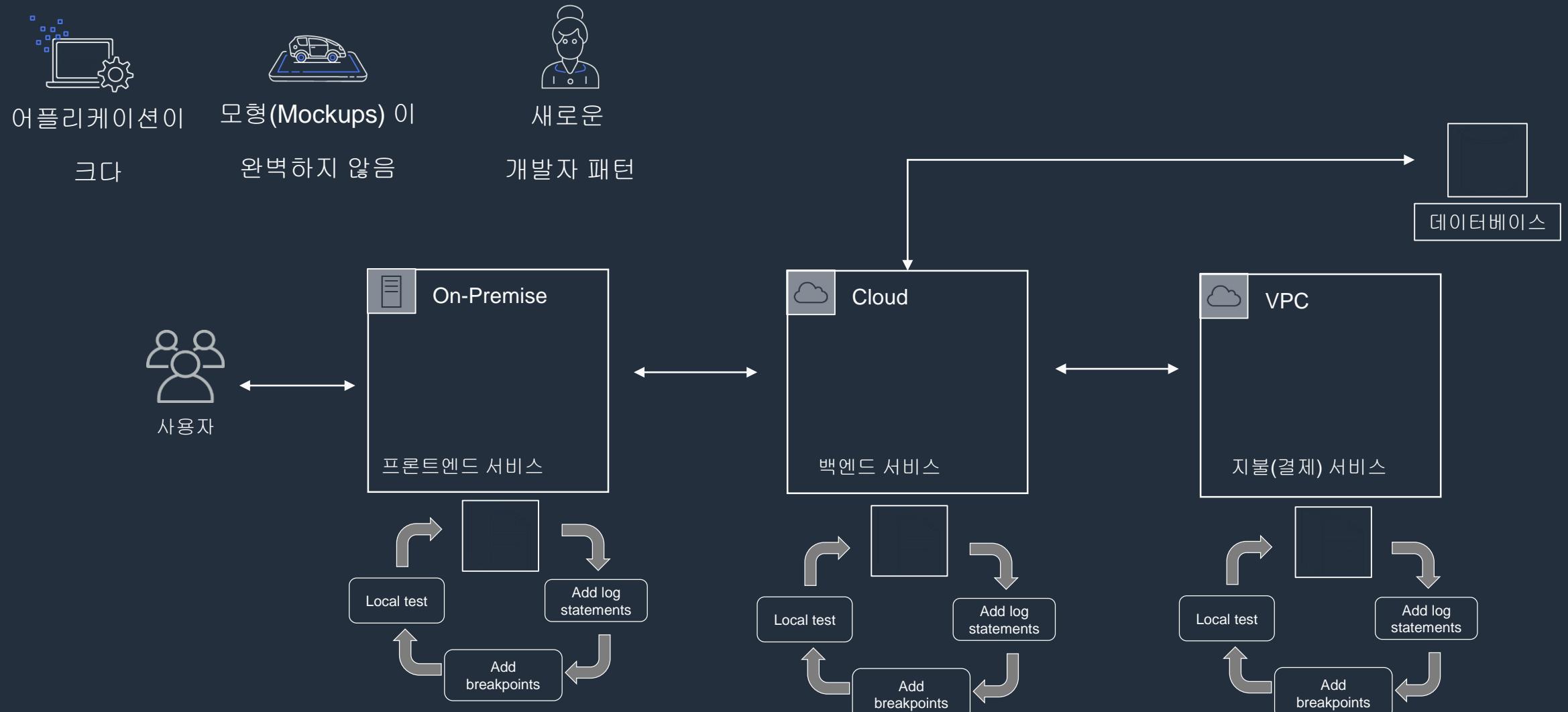


© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

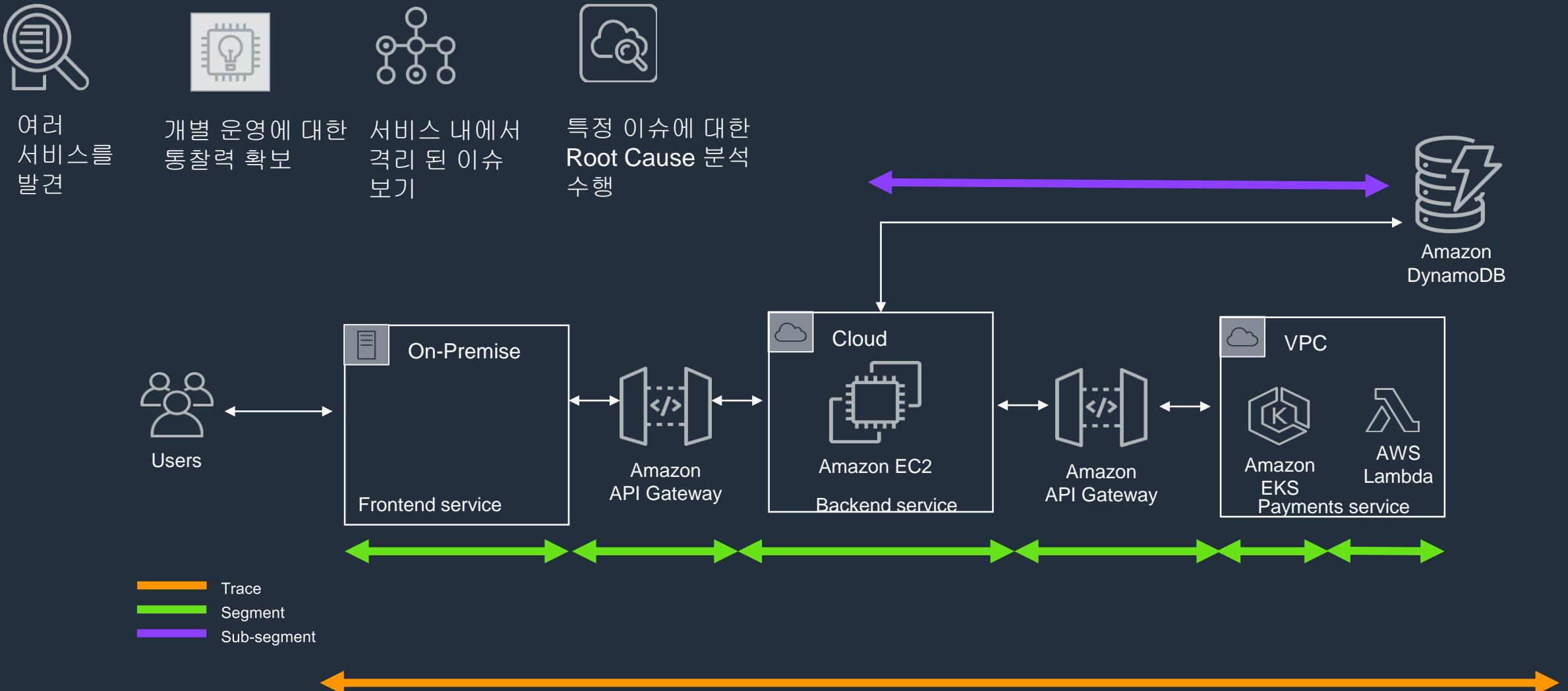
전통적인 디버깅



현대적(Modern) 애플리케이션은 로컬에서 디버깅하기 어려움



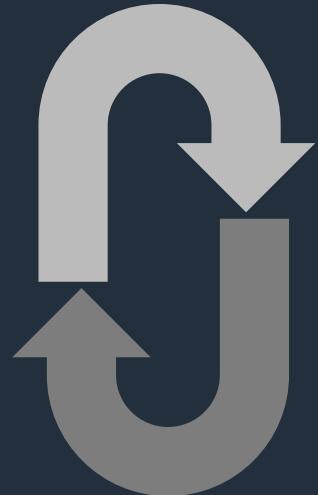
현대적(Modern) 애플리케이션을 위한 추적



현대적(Modern) 애플리케이션을 위해 제작된 AWS X-Ray



문제를 신속하게 분석
및 디버깅



개별 서비스들의 End-
to-End 보기



고객 영향도 파악



Cloud agnostic
(Serverless 지원)

X-Ray 서비스 동작 방법

TRACE REQUESTS



X-Ray collects data about the request from each of the underlying applications services it passes through

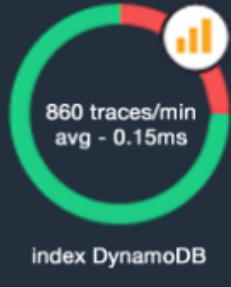
RECORD TRACES



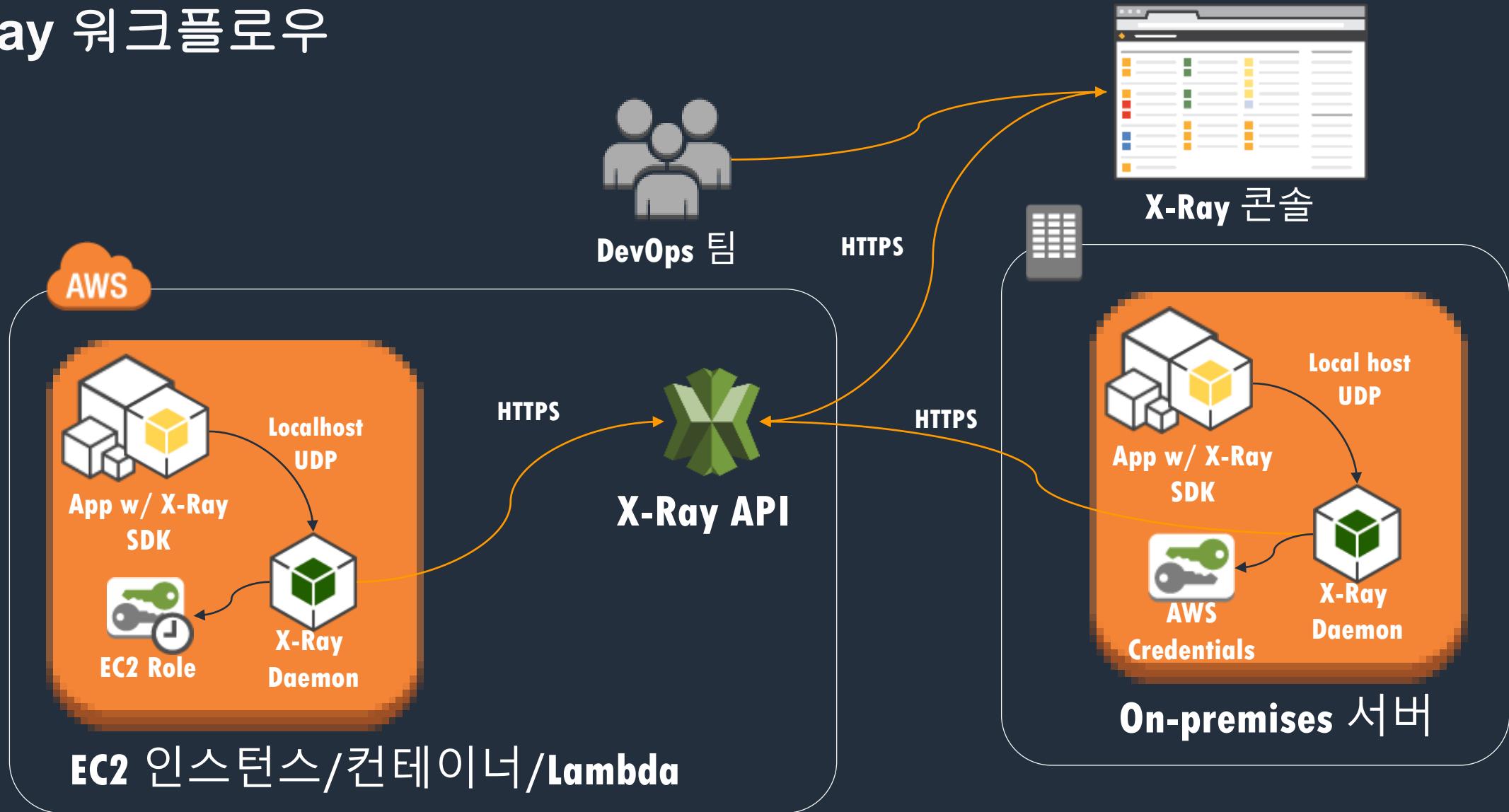
VIEW SERVICE MAP



ANALYZE ISSUES



X-Ray 워크플로우



X-Ray SDK

Java, .NET, .NET Core, Python, Ruby, Go 및 **Node.js**에 사용 가능

Metadata 를 자동으로 캡처하는 필터를 추가:

- AWS SDK를 사용하는 AWS 서비스
- HTTP 및 HTTPS를 통한 Non-AWS 서비스
- 데이터베이스 (MySQL, PostgreSQL 및 Amazon DynamoDB)
- Queues (Amazon SQS)

Request에 대한 메타 데이터를 Logging 하기 위해 애플리케이션 코드를 수동으로 변조하지 않고도 빠르게 시작할 수 있음

Source on GitHub under <https://github.com/aws?q=xray-sdk>



X-Ray Daemon

UDP를 통해 SDK에서 데이터를 수신하고 로컬 버퍼 역할을 수행. 매 초마다 또는 로컬 버퍼가 채워질 때 데이터가 백엔드로 전송 및 비워짐

Amazon Linux AMI, RHEL, Ubuntu, macOS 및 Windows에서 사용 가능

AWS Lambda에 사전 설치되어 있음

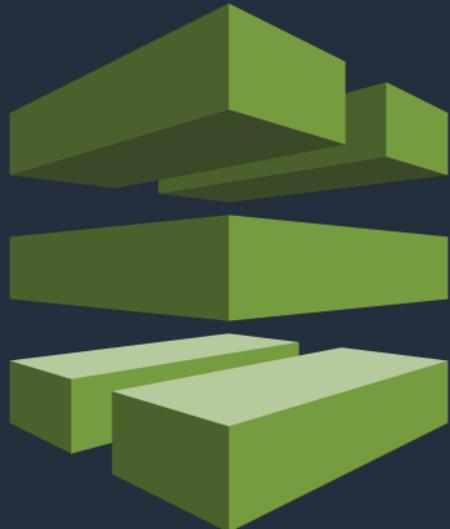
AWS 자격 증명이 제공되는 한 어디에서나 실행 가능 (예 : EC2, Amazon ECS, 온프레미스, 개발자 컴퓨터 등)

Source on GitHub under <https://github.com/aws/aws-xray-daemon>



파이프라인 관리

AWS CodePipeline

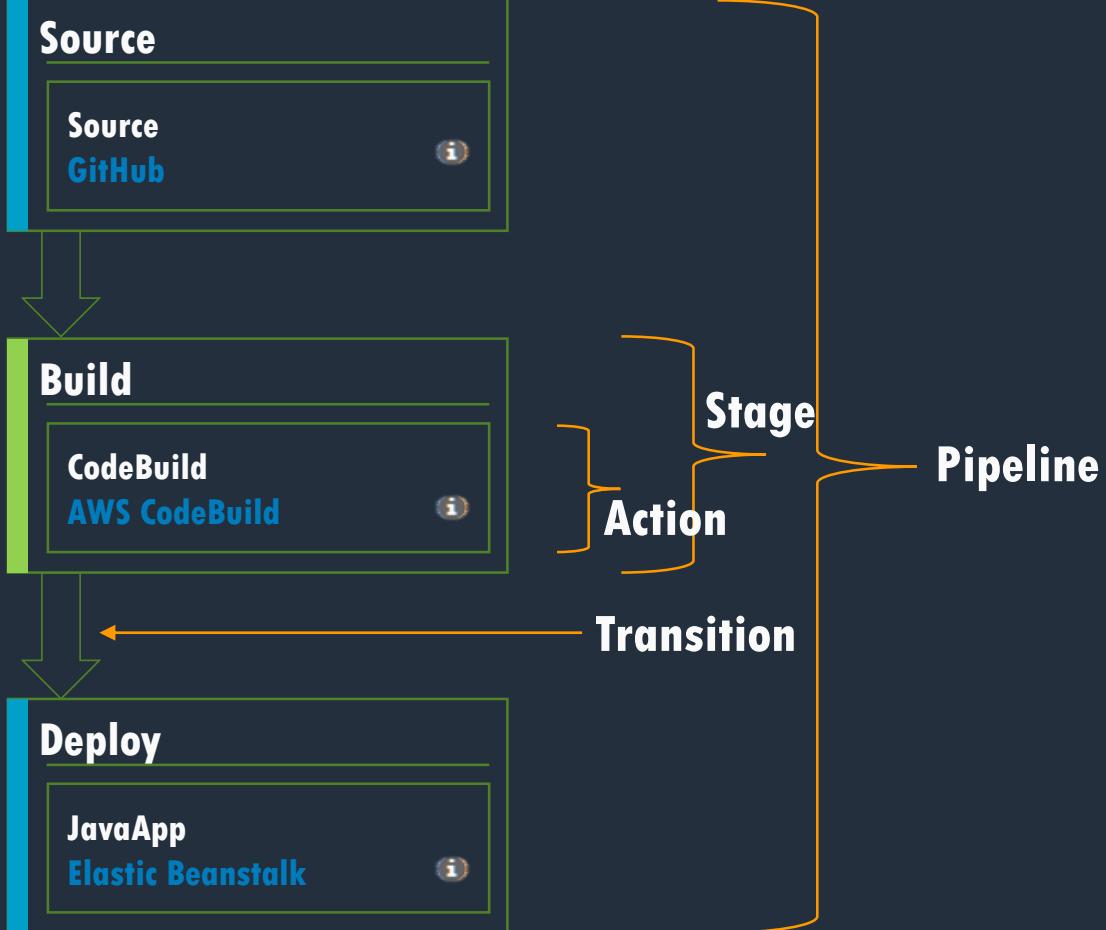


- 빠르고 신뢰성 있는 어플리케이션 업데이트를 위한 지속적인 전달 (**Continuous Delivery**) 서비스
- 소프트웨어 릴리스 프로세스 모델링 및 시각화
- 코드가 변경 될 때마다 코드를 빌드, 테스트 및 배포
- 타사 도구 (**3rd-party**) 및 **AWS** 와 통합



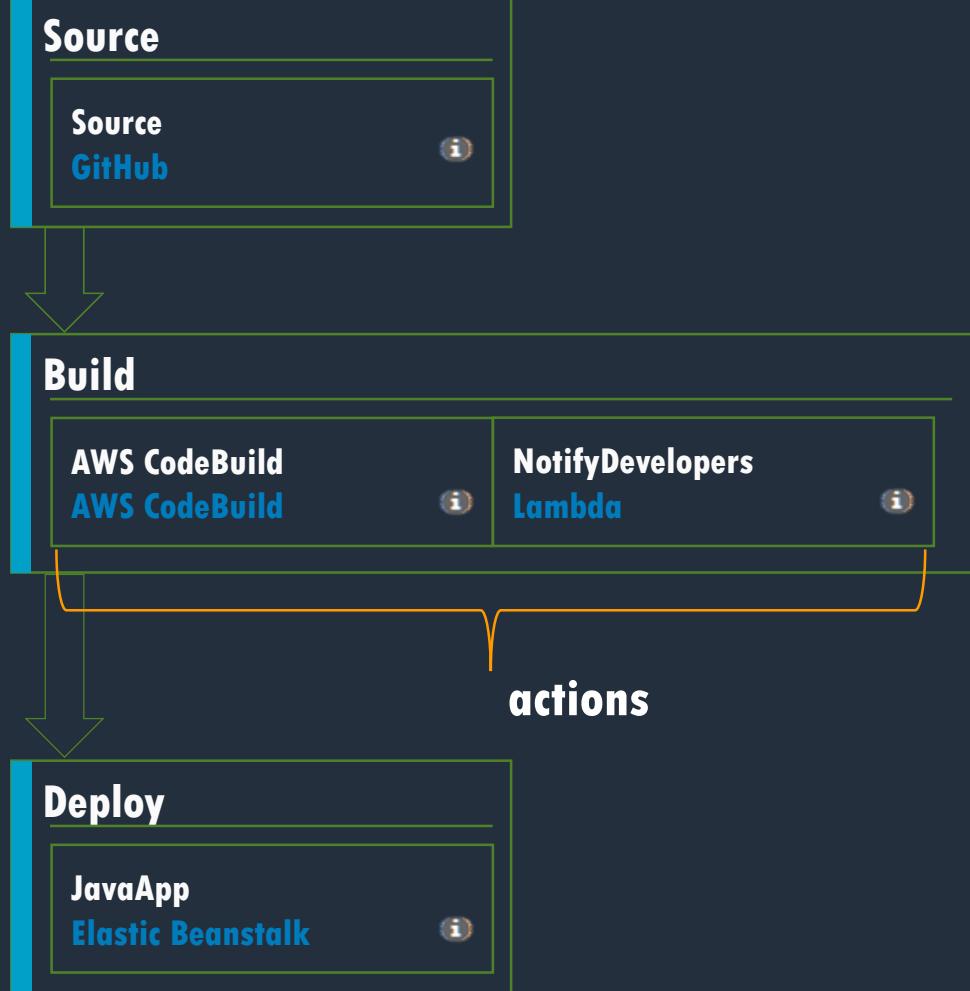
AWS CodePipeline

MyApplication



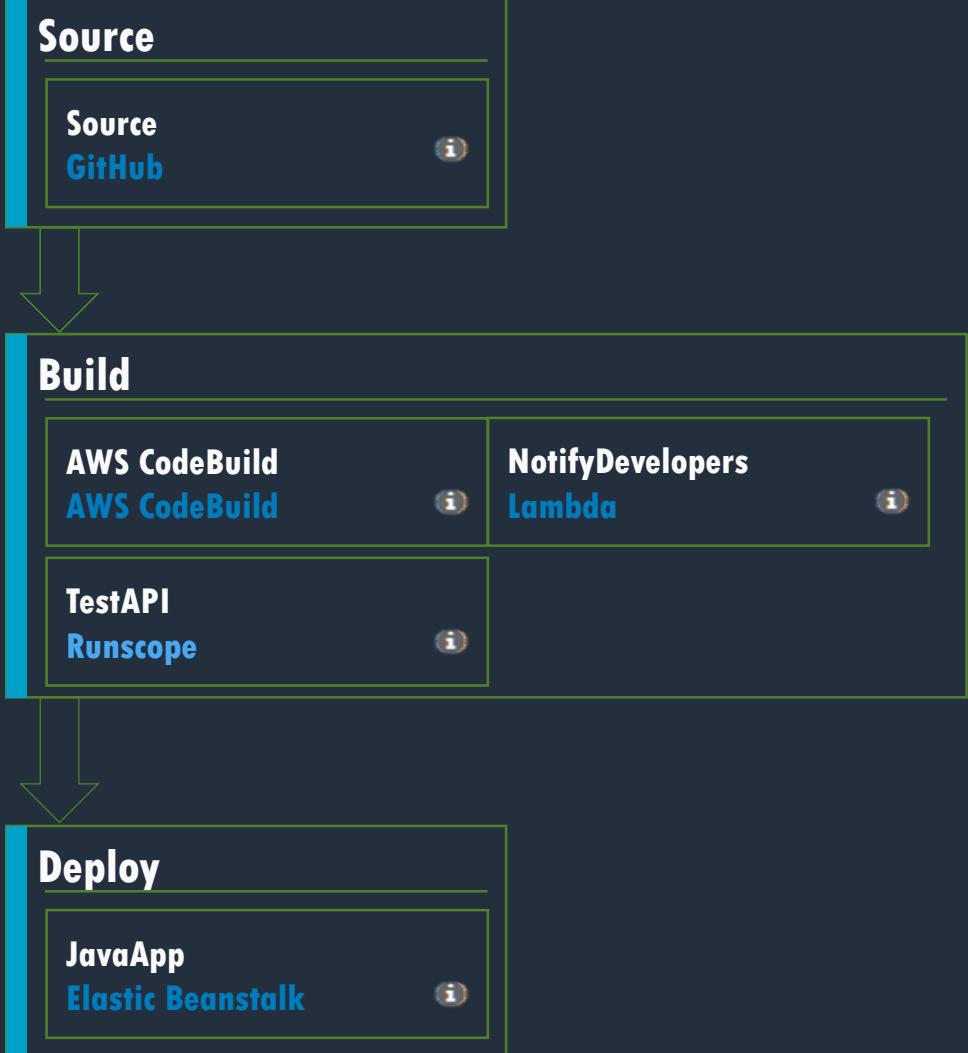


MyApplication





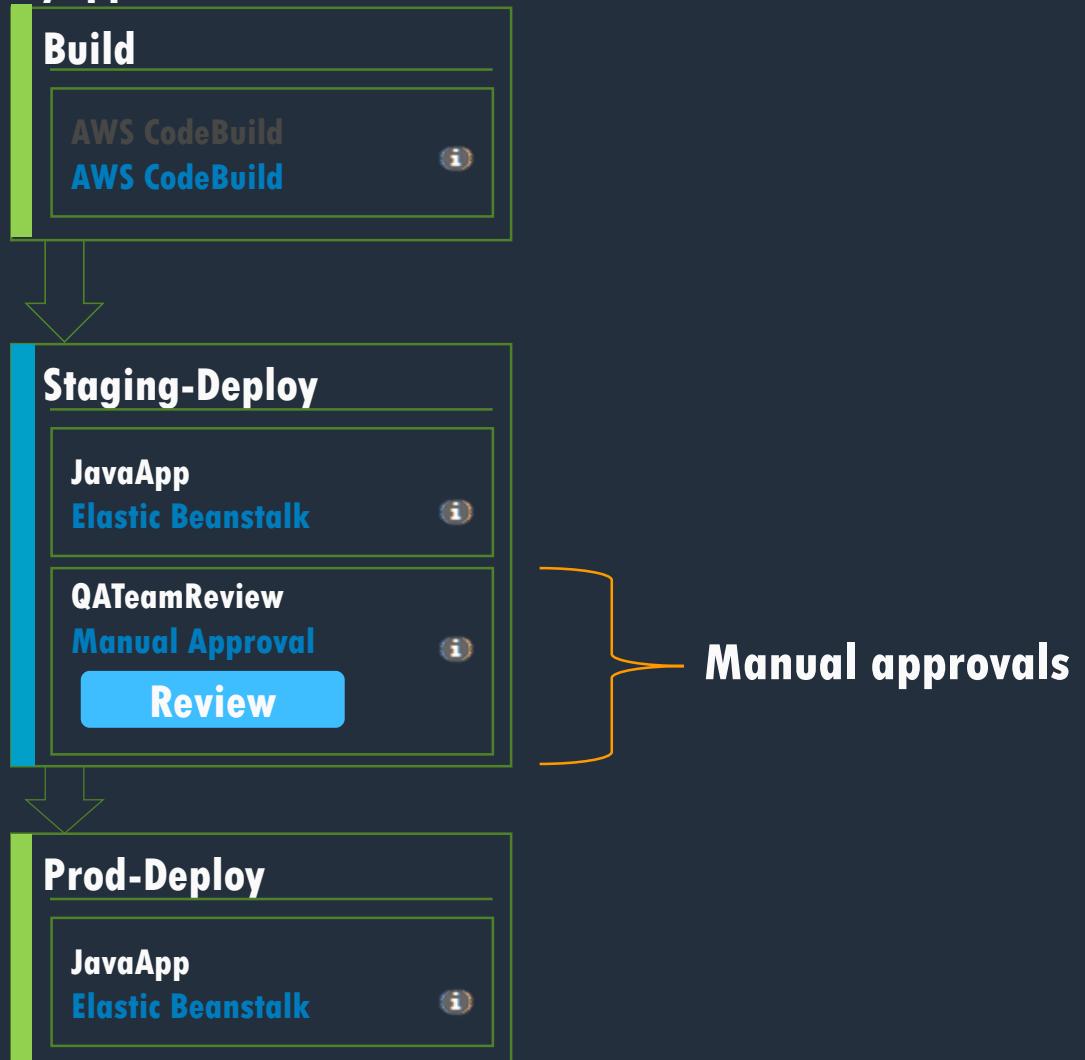
MyApplication



Sequential actions



MyApplication



Manual approvals

데모(실습)

<https://bit.ly/38Hfl80>



Appendix.

- 프로젝트 관리

AWS CodeStar



- AWS에서 애플리케이션을 신속하게 개발, 빌드 및 배포
- 몇 분 안에 AWS에서 개발 시작
- 팀 전체에서 안전하게 작업
- 소프트웨어 제공을 쉽게 관리
- 다양한 프로젝트 템플릿 중에서 선택

AWS CodeStar

EC2, AWS Lambda 및 Elastic Beanstalk 용 프로젝트 템플릿

AWS CodeStar

Select Template Setup Tools Start Coding

Select an application template to get started. [Help me choose](#)

Refine by

- Application category
 - Web application
 - Web service
- Programming languages
 - Ruby
 - Node.js
 - Java
 - Python
 - PHP
 - HTML 5
- AWS services
 - AWS Elastic Beanstalk
 - Amazon EC2
 - AWS Lambda

 Ruby on Rails Web application  AWS Elastic Beanstalk (runs in a managed application environment)	 Ruby on Rails Web application  Amazon EC2 (runs on virtual servers that you manage)	 Java Spring Web application  AWS Elastic Beanstalk (runs in a managed application environment)	 Java Spring Web application  Amazon EC2 (runs on virtual servers that you manage)
 Node.js Web application  AWS Lambda (running serverless)	 Node.js Web application  AWS Elastic Beanstalk (runs in a managed application environment)	 Node.js Web application  Amazon EC2 (runs on virtual servers that you manage)	 Python (Django) Web application  AWS Elastic Beanstalk (runs in a managed application environment)
 Python (Django) Web application	 Express.js Web application	 Express.js Web application	 PHP (Laravel) Web application



AWS CodeStar

소스 제어 제공자 (**Source Control Provider**) 선택

AWS CodeStar ▶ Create project

Project name
TwoPizzas

Project ID ⓘ Edit
twopizzas

Which repository do you want to use?
AWS CodeStar will store the project's source code with the service you choose here.

 AWS CodeCommit
Highly available Git source control from AWS.
Includes encryption, IAM integration, and more.

 GitHub
Creates a GitHub source repository for this project. Requires an existing GitHub account.

Repository name
TwoPizzas



AWS CodeStar

사전 구성된 지속적 전달(Continuous Delivery) 툴체인

AWS CodeStar

Select Template Setup Tools Start Coding

Project name: AWS SF Summit | Project ID: aws-sf-summit | Edit

AWS CodeStar includes all of the tools and services you need for a development project.
This project includes:

Source > Build > Test > Deploy > Environment

Release Automation: AWS CodePipeline

Language: Node.js Build: AWS CodeBuild Deployment: AWS CodeFormation Serverless: AWS Lambda

Source Control: AWS CodeCommit Monitoring: Amazon CloudWatch

AWS Developer Hub would like permission to administer AWS resources on your behalf. [Learn more](#)

Previous Create Project



AWS CodeStar

선호하는 IDE에 쉽게 연결

AWS CodeStar ▶ Create project

Select template Set up tools Start coding

Pick how you want to edit your code

 **AWS Cloud9**
Edit your AWS CodeStar project code with a cloud-based IDE that includes a command line interface. [More info](#)

 **Command line tools**
Edit AWS CodeStar project code by connecting directly to your project's Git source repository.

 **Eclipse**
Configure the AWS Toolkit for Eclipse to edit your AWS CodeStar project code in Eclipse.

 **Visual Studio**
Configure the AWS Toolkit for Visual Studio to edit your CodeStar project code in Microsoft Visual Studio 2015 and later.



AWS CodeStar

몇 번의 클릭만으로 안전한 팀 액세스 설정

AWS CodeStar ▶ AWS SF Summit Demo

Project Team

Manage users and permissions in your project.

Add team member

Display Name	Email Address	Project Role	Remote Access
user1	Type email address (Optional)	Viewer	<input type="checkbox"/>
user1		Contributor	
user1		Owner	

Add

Team member list (3)

	Name	Email	Role	Remote Access	
C	cdhull [You]	cdhull@xyz.com	Owner	Add Public SSH key	Remove Edit
A	Alex		Contributor	Not Granted	Remove Edit
L	Iuisarias		Viewer	Not Granted	Remove Edit

Search team members

aws

AWS CodeStar

통합 대시 보드 - 전달 파이프 라인 및 문제 추적 관리

The screenshot displays the AWS CodeStar integrated dashboard for the 'AWS SF Summit Demo' project. On the left, a sidebar navigation bar includes links for Dashboard, Code, Build, Deploy, Pipeline, Connect, Team, Extensions, and Project. The main interface features a JIRA board view titled 'My First Project' under 'MFP board' and 'MFP Sprint 1'. The board lists four issues:

Key	Type	Status	Priority	Issue Summary
MFP-1	Story	To Do	Medium	Clone the repo
MFP-2	Story	To Do	Medium	Make a change to the sample code
MFP-3	Story	To Do	Medium	Push the change to the project repo
MFP-4	Story	To Do	Medium	Watch my project automatically pick up and deploy the change

Below the board are sections for 'Unit Testing Code Samples' and 'User Gamification Feature Testing Samples', both with edit options. A code snippet is shown in the testing section:

```
var increment = function (num) {
    return num++;
};
```

On the right, a 'Continuous deployment' pipeline is shown with three stages: Source, Build, and Deploy. The Source stage (CodeCommit) is green ('Succeeded'). The Build stage (CodeBuild) is green ('Succeeded'). The Deploy stage (CloudFormation) has failed ('Failed') at 4/3/2017, 10:57:11 AM.



AWS CodeStar

통합 대시 보드 - 어플리케이션 활동 및 Commit history

Team

Extensions

Project

Application activity

CPUUtilization

0.416
0.17
0.032

16:00 17:00 18:00 19:00 20:04 20:08 21:00 22:00 23:00 00:00 01:00 02:00 03:00 04:00 05:00 06:00 07:00 08:00 09:00 10:00 11:00 12:00 13:00 14:00 15:00 16:00

CPUUtilization

AWS CloudWatch

AWS CloudWatch details

Commit history: sample-project-Repo

Master

AD Initial commit of sample code made during project creation in AWS DevHub AWS DevHub committed 1 month ago 780f7b1

Connect AWS CodeCommit details

Application endpoint(s)

<http://dh-w-eben-a4juvih36y.eijmpiy2jy.us-east-1.el...>

Continuous deployment

AWS CodePipeline

Source
12/20/2016, 1:53:45 PM ApplicationSource CodeCommit Succeeded

Commit history

Application
12/20/2016, 1:55:25 PM Deploy CodeDeploy Succeeded

Deploy history Hosts(1)

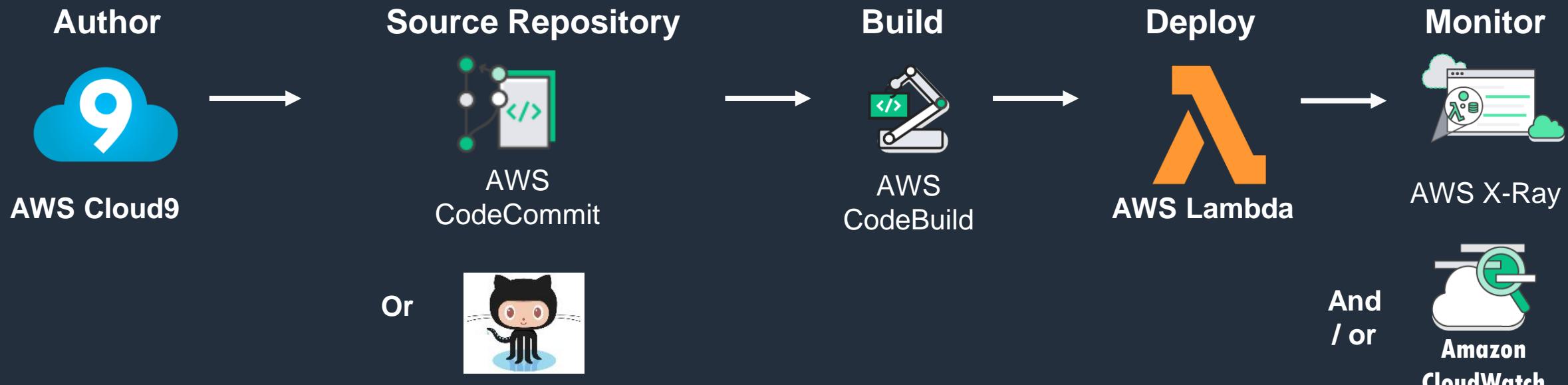
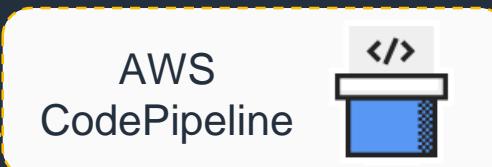


Appendix.

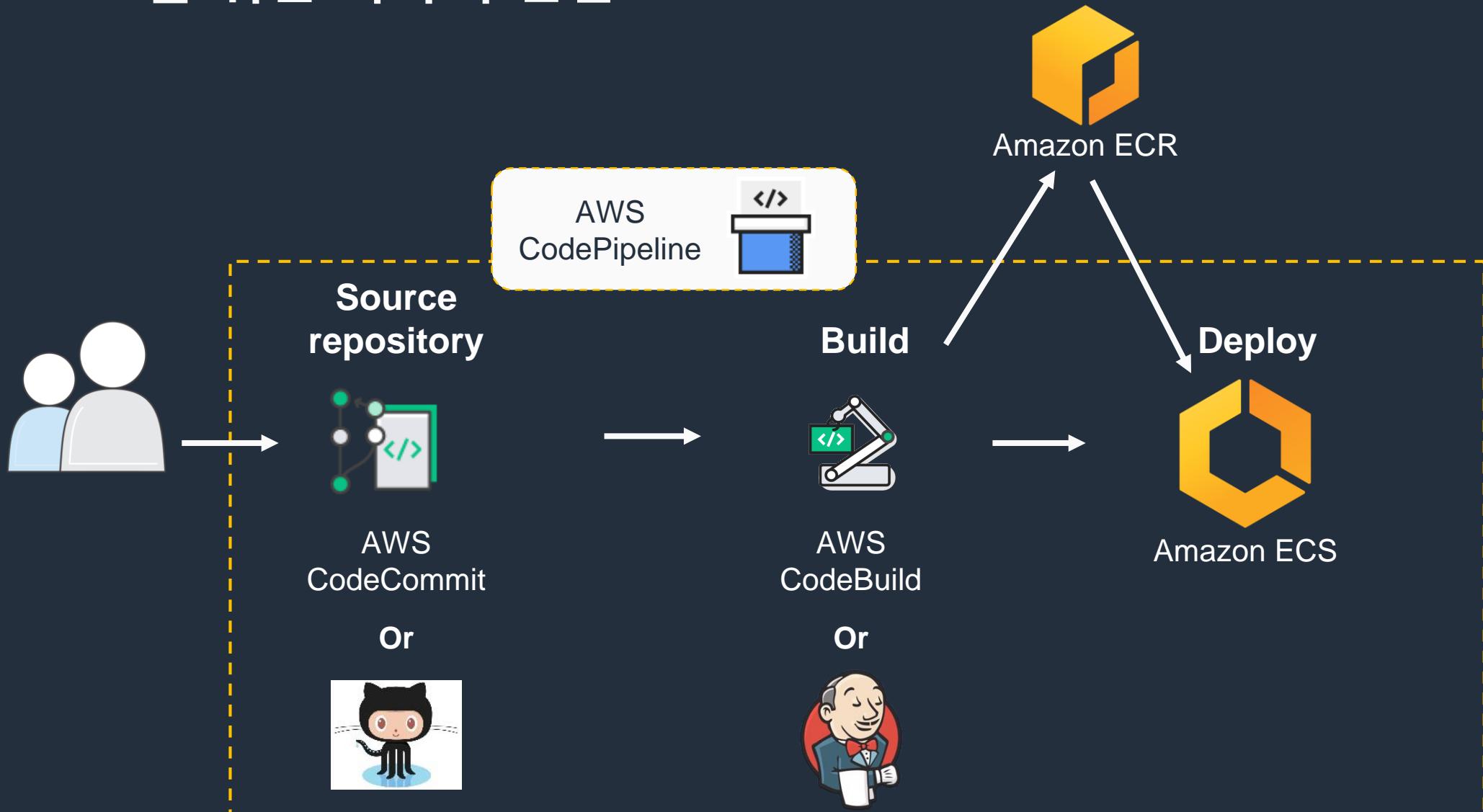
- CI/CD Use Case

Serverless 어플리케이션을 위한 지속적 전달

AWS CodeStar



Container를 위한 지속적 전달



감사합니다!



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.