

오픈소스로 여는 뉴노멀

# 2020 공개SW 페스티벌

# Opensource Project DevOps Art

beNX  
DevOps 엔지니어  
송주영  
(jupitersong47@gmail.com)

# ▶ DevOps engineer & AWS Container hero

Developer / Community / Heroes / ...

## Juyoung Song





**Juyoung Song, DevOps Engineer at beNX**

Seoul, Korea

Hero since 2019

Juyoung Song is a DevOps Engineer at beNX. He is currently in charge of transforming the legacy-cloud systems into modern cloud architecture to bring global stars such as BTS and millions of fans together in the digital sphere.

Previously he was at Samsung Electronics as a DevOps Engineer where he shared best practices and migration of modern cloud architectures. Samsung Account is an account platform which serves more than 900,000,000 users, and he contributed to the non-stop migration of Samsung Account from on-premises to AWS cloud.

Juyoung has spoken regularly at AWS-organized events such as AWS Container Day, AWS Summit, and [This is My Architecture](#). Furthermore, he organized and spoke at various Meetups like [AWS Korea User Group](#) and [DevOps Korea](#), about topics such as ECS and Fargate, and its DevOps best practices. He has carried on his expertise to writing, by producing written content for blogs and IT magazines in Korea. He is interested in building hyper-scale DevOps environments for containers using AWS CodeBuild, Terraform, and various open-source tools. His goal is to grow from DevOps engineer to DevOps producer, and ultimately DevOps Artist to maximize performance, work-emotion, cost, tools and methodology to build cloud-native services.

[Connect with Juyoung](#)

- DevOps engineer at beNX (Weverse)
- AWS Container hero
- AWS 한국사용자그룹 DevOps organizer
- Project DevOps Art administrator (<https://github.com/DevopsArtFactory>)
- Youtube (<https://youtube.com/c/devopsart>)
- DevOps workshop (<https://devops-art-factory.gitbook.io/devops-workshop>) 2

aws 송주영

About 1,720 results (0.36 seconds)

aws.amazon.com > blogs > korea > g... > Translate this page

**한국의 AWS Heroes를 소개합니다! (2020년 5월) | Amazon ...**

Mar 7, 2019 - 국내 최초 컨테이너 히어로인 송주영님은 유명 아이돌 그룹인 BTS의 온라인 팬서비스를 담당하는 beNX의 데브옵스 엔지니어로 일하고 있습니다.

Videos



**Samsung Knox 및 Connect의 AWS 기반 콘테이너 활용 사례 ...**

Amazon Web Services  
YouTube - May 9, 2017



**데브옵스 아티스트 송주영님과 함께 - AWS Hero 특집 :: 차니의 ...**

Channy Yun - 윤석찬  
YouTube - Mar 25, 2020



**DevOps Art 1년 부채부터 10년 부채까지 - 송주영**

DevOps Korea  
YouTube - Sep 13, 2019

www.zdnet.co.kr > view > Translate this page

**AWS히어로 개발자 "업무성과, 강압 아닌 자유에서 나와 ...**

Dec 4, 2019 - AWS 히어로로 선정된 비엔엑스(beNX)의 송주영 데브옵스 엔지니어는 2일 미국 라스베이거스에서 열린 'AWS 리인벤트 2019' 현장에서 감동적인 ...

Images for aws 송주영

상상전자 continuous integration 데브옵스 samsung ecs aws container docker 컨테이너 >






## ▶ Introduction

- What is DevOps ?
- Project DevOps Art
- Deployment best practices ( feat.goployer )
- Testing best practices ( feat. bigshot )

▶ What is DevOps ?

# DevOps

What is DevOps ?

▶ DevOps 를 이루는 5가지 철학

## 5가지 철학

문화  
(Culture)

DevOps를 통해 하나의 문화를 만들어갑니다.

자동화  
(Automation)

자동화를 통해 효율성과 빠른 속도를 지향합니다.

측정  
(Measurement)

지표를 측정하여 지속적으로 개선해 나갑니다.

공유  
(Sharing)

공유를 통해 함께 발전해 나갑니다.

축적  
(File up & Pile

기록을 축적하여 자산을 만들어 나갑니다.

## ▶ What is DevOps ?

**DevOps** 는 어떤 요구사항을 효율적으로 만족시키기 위하여,  
일을 **자동화**하며 변경사항 지표들을 **측정**하고 **공유**하고,  
이 모든 결과물들을 지속적으로 **추적**해 나아가는 **문화**를 만들어가는  
철학, 방법론, 기술

▶ What is Project DevOps Art ?

# Project DevOps Art

From Tech to Art

## ▶ Project DevOps Art



## DevOps Art

DevOps 철학의 올바른 개념적 이해와 철학에 기반한 이상적인 구현을 위한 프로젝트

DevOps 의 목적인 업무 속도와 효율화를 위해, 코드를 공유하고 강의하며, 오픈소스를 기획 및 개발하고 있습니다.

- Sharing Infrastructure as Code for best practices
- CLI for automation
- Opensource deployment tool
- Opensource testing tool
- Online workshop

Github: <https://github.com/DevOpsArtFactory>

▶ What is DevOps ?

# Deployment best practices

Goployer

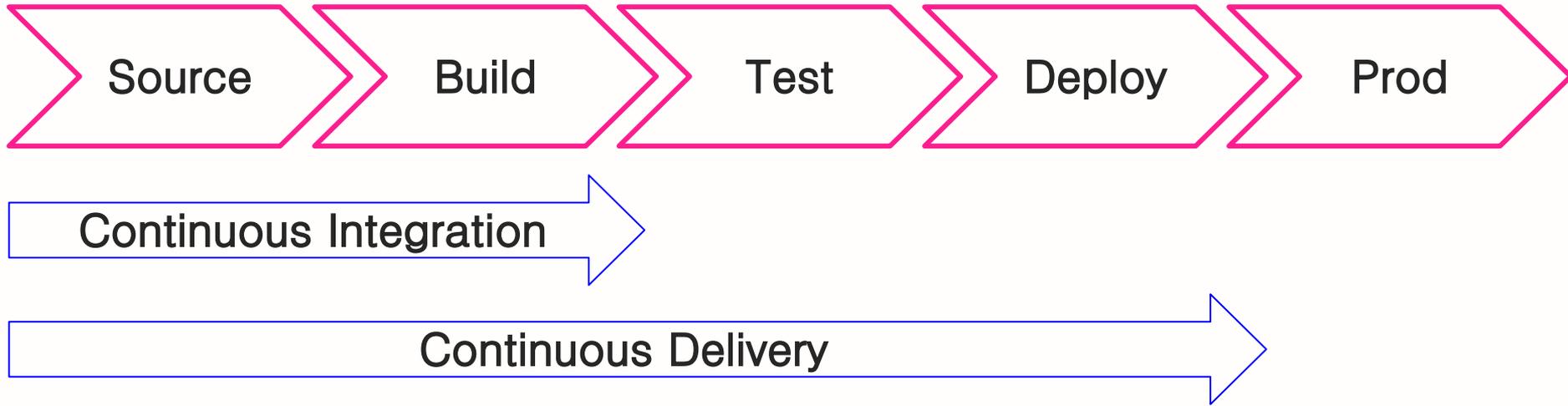
(feat. Declarative Infrastructure as Code)

Project goployer: <https://github.com/DevopsArtFactory/goployer>

## ▶ Continuous Integration



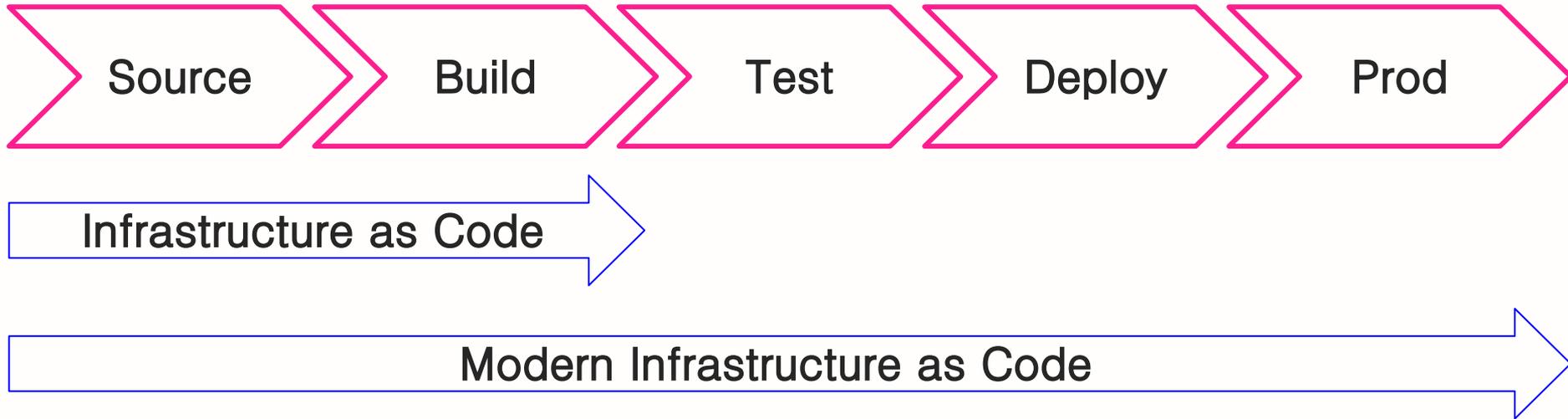
## ▶ Continuous Delivery



## ▶ Infrastructure as Code

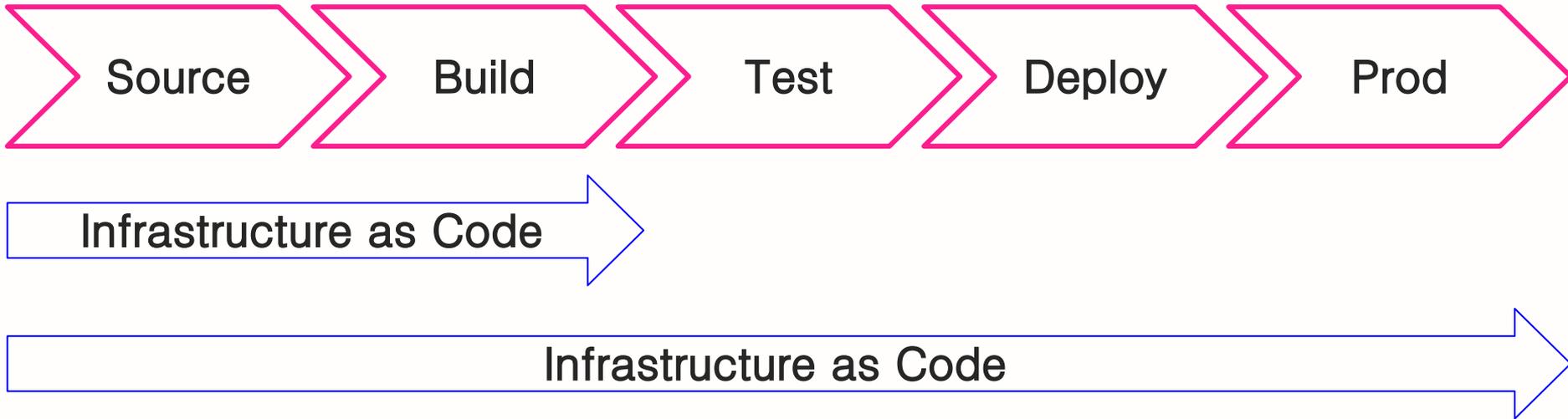


## ▶ S/W Cycle & Infrastructure as Code



- Sourcecode repo
- Accounts
- Test
- Deploy
- Monitoring

▶ S/W Cycle & Infrastructure as Code



Imperative vs Declarative  
(명령형) (선언형)

▶ <https://goployer.dev> – Opensource deployment tool

**goployer**    🔄    🌐    📄    문서    korean ▾

빠르고, 강력하고, 사용하기 간단한.  
**AWS 배포 도구**

Goployer 설치 📄    시작하기 ➔

Goployer는 AWS 오토스케일링, 로드 밸런싱을 활용하여 전체 배포 프로세스를 관리합니다.

**프로덕션 수준**  
Goployer는 오토스케일링을 활용한 배포에 있어서 대부분의 기능을 제공합니다.  
충분한 테스트 후에는 프로덕션 수준에서 사용 가능합니다.

**Contribution 환영**  
깃허브를 통해 [Pull Request](#)를 받고 있습니다.  
어떠한 의견이라도 저희는 환영입니다.

**어플리케이션 개발 집중**  
더 이상 배포 과정은 신경쓰실 필요가 없습니다.  
개발에 집중하시고 배포는 Goployer로 하시면 됩니다.

**beNX**

## ▶ Best practices for deployment

### Best practices

#### Immutable Infrastructure

#### Deployment as Code

#### Measurement

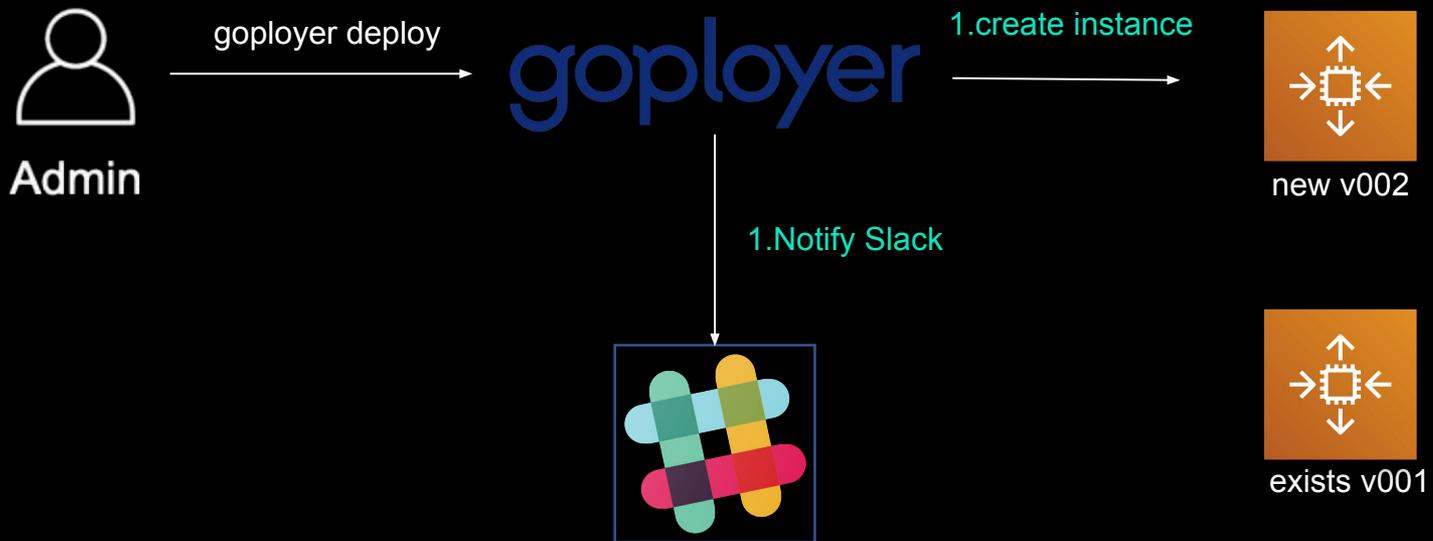
#### Test

#### Cost effective

- Servers are never modified after they're deployed.
- If server has some problem, terminate it!
- If something needs to be updated, do deploy!
- Do troubleshooting !

But do not change something in server

- Ensure each phase is the same
- Create and use golden AMI by Packer

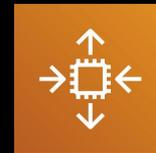




goployer deploy

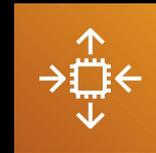
goployer

1.create instance

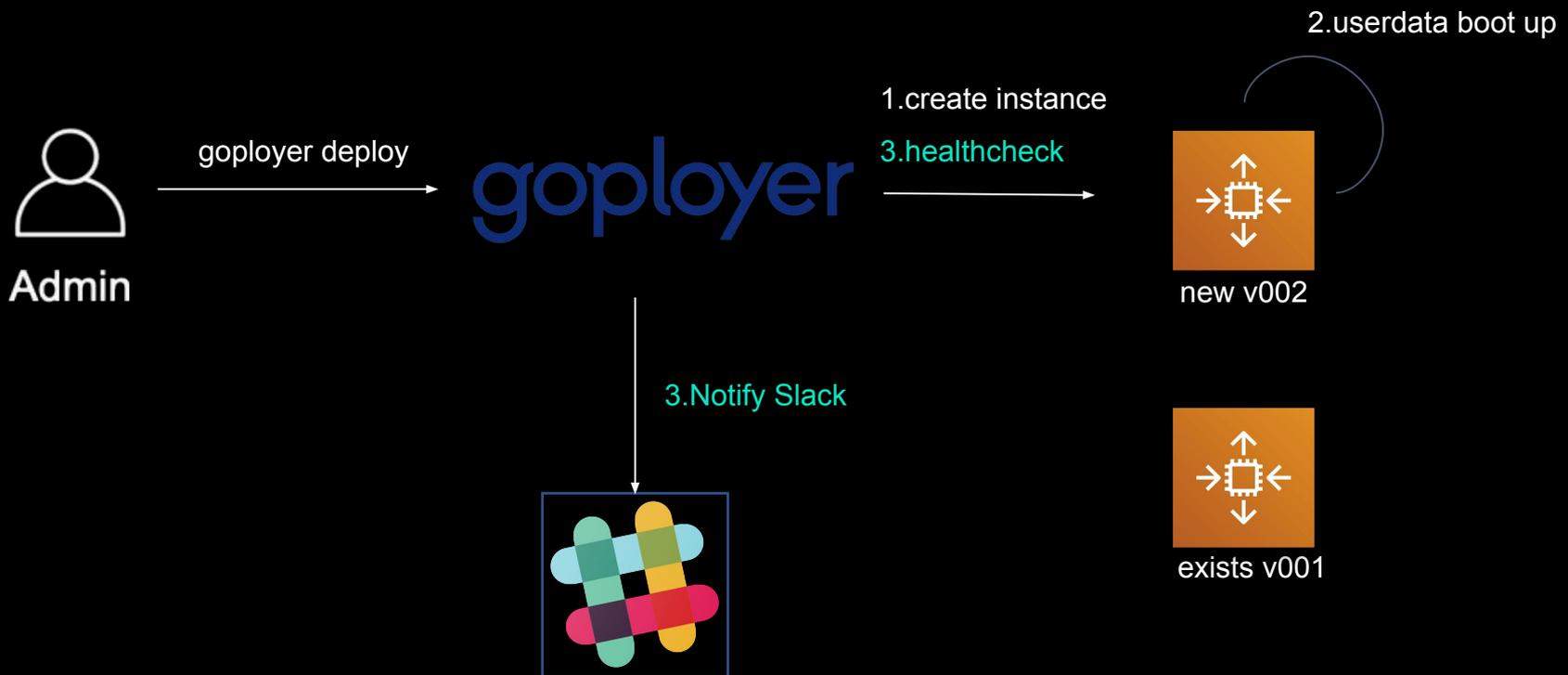


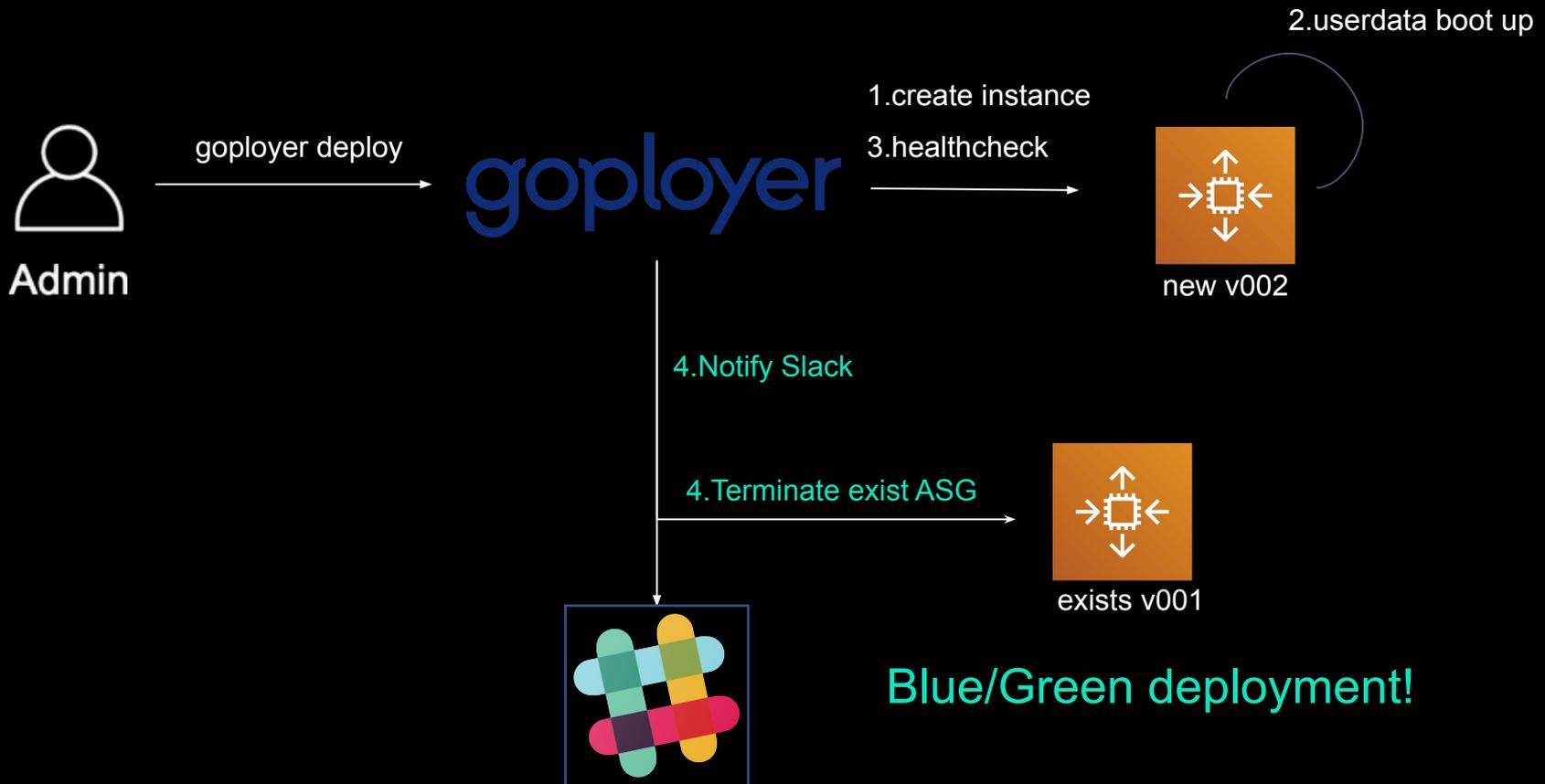
new v002

2.userdata boot up



exists v001





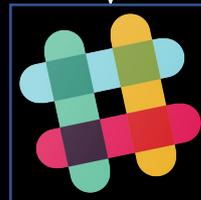


goployer deploy

goployer



5. Notify Slack



## ▶ Best practices for deployment

### Best practices

Immutable Infrastructure

Deployment as Code

Measurement

Test

Cost effective

```
39 tags:
40   - project=test
41   - repo=hello-deploy
42
43 stacks:
44   - stack: artd
45     polling_interval: 30s
46     account: dev
47     env: dev
48     replacement_type: BlueGreen
49     iam_instance_profile: app-hello-profile
50     ebs_optimized: true
51     block_devices:
52       - device_name: /dev/xvda
53         volume_size: 15
54         volume_type: "gp2"
55       - device_name: /dev/xvdb
56         volume_type: "st1"
57         volume_size: 500
58     capacity:
59       min: 1
60       max: 2
61       desired: 1
62     autoscaling: *autoscaling_policy
63     alarms: *autoscaling_alarms
64     lifecycle_callbacks:
65       pre_terminate_past_cluster:
66         - service hello stop
67
68     regions:
69       - region: ap-northeast-2
70         instance_type: t3.medium
71         ssh_key: test-master-key
72         ami_id: ami-01288945bd24ed49a
73         use_public_subnets: true
74         vpc: vpc-artd_apnortheast2
75         detailed_monitoring_enabled: false
76         security_groups:
77           - hello-artd_apnortheast2
78           - default-artd_apnortheast2
79         healthcheck_target_group: hello-artdapne2-ext
80         availability_zones:
81           - ap-northeast-2a
82           - ap-northeast-2b
83           - ap-northeast-2c
84         target_groups:
85           - hello-artdapne2-ext
```

## ▶ Best practices for deployment

### Best practices

Immutable Infrastructure

Deployment as Code

Measurement

Test

Cost effective

- If move it, measure it
- Get insight from everything
- metrics.yaml
- AWS NoSQL Service DynamoDB
  - Deployment info: code, date
  - Metric for server: Uptime,
  - Stats: RequestCounts,

## ▶ Best practices for deployment

<input type="checkbox"/>	identific	deployment_statu	config	release-notes
<input type="checkbox"/>	hello-v	terminated	{"manifest":"deployments/hello.yml","manifest_s3_region":"","ami":"ami-0c8916..."}	By goployer
<input type="checkbox"/>	hello-v	deployed	{"manifest":"deployments/hello.yml","manifest_s3_region":"","ami":"ami-0c8916..."}	By goployer
<input type="checkbox"/>	hello-v	terminated	{"manifest":"deployments/hello.yml","manifest_s3_region":"","ami":"ami-0c8916..."}	By goployer

terminated_date	statistics_record_time	start_date	deployed_date	uptime_hour	uptime_minute
2020-08-26T07:46:12Z	2020-08-26T07:46:13Z	2020-08-24T07:44:44Z	2020-08-24T07:47:17Z	47.982126	2878.927542
2020-08-24T07:49:51Z	2020-08-24T07:49:52Z	2020-08-20T07:12:47Z	2020-08-20T07:15:20Z	96.575374	5794.522468
2020-08-20T07:18:25Z	2020-08-20T07:18:26Z	2020-08-19T14:17:32Z	2020-08-19T14:20:06Z	16.972187	1018.331204
2020-08-14T01:57:24Z	2020-08-14T01:57:24Z	2020-08-14T00:45:02Z	2020-08-14T00:47:36Z	1.163491	69.809457

2020-08-07T10:00:00Z Number : 206330.3375  
 2020-08-07T11:00:00Z Number : 108170.0625  
 2020-08-07T12:00:00Z Number : 395182.6625  
 2020-08-07T13:00:00Z Number : 266578.0125  
 2020-08-07T14:00:00Z Number : 144047.7625

2020-08-25T00:00:00Z Number : 26108.0375  
 2020-08-25T01:00:00Z Number : 25123.55  
 2020-08-25T02:00:00Z Number : 25553.325  
 2020-08-25T03:00:00Z Number : 36676.825  
 2020-08-25T04:00:00Z Number : 32434.390554  
 2020-08-25T05:00:00Z Number : 46389.233862  
 total Number : 1027216.305707

## ▶ Best practices for deployment

### Best practices

Immutable Infrastructure

Deployment as Code

Measurement

**Test**

Cost effective

- Enable/Disable automate load test
- Integrated with Vegeta
  - <https://github.com/tsenart/vegeta>
  - Simple http loadtest tool
  - Support Go library

## ▶ Best practices for deployment

```
stacks:
- stack: artd
  polling_interval: 30s
  account: dev
  env: dev
  assume_role: ""
  replacement_type: BlueGreen
  iam_instance_profile: 'app-hello-profile'
  ansible_tags: all
  ebs_optimized: true
  api_test_enabled: true
  instance_market_options:
    market_type: spot
    spot_options:
      block_duration_minutes: 180
      instance_interruption_behavior: terminate # terminate / stop / hibernate
      max_price: 0.3
      spot_instance_type: one-time # one-time or persistent
  block_devices:
  - device_name: /dev/xvda
    volume_size: 10
    volume_type: "gp2"
  - device_name: /dev/xvdb
    volume_size: 500
    volume_type: "st1"
  capacity:
    min: 1
    max: 2
    desired: 1
  autoscaling: *autoscaling_policy
  alarms: *autoscaling_alarms
  lifecycle_callbacks:
    pre_terminate_past_cluster:
      - service hello stop
```

```
90
99  api_test_template:
100  name: api-test
101  duration: 5s
102  request_per_second: 10
103  apis:
104    - method: GET
105      url: https://example.com
106    - method: POST
107      url: https://example.com/post
108      body:
109        - id=1234
110        - username=art
111        - test=test
```

API: <https://hello-API URL Endpoint>

Duration: 4.90s

Wait: 10.00ms

Requests: 50

Rate: 10.20

Throughput: 10.18

Success: 1.00

Latency P99: 64.00ms

[Show less](#)

API: <https://hello-API URL Endpoint/adduser>

Duration: 4.90s

Wait: 11.00ms

Requests: 50

Rate: 10.20

Throughput: 0.00

Success: 0.00

Latency P99: 65.00ms

[Show less](#)

## ▶ Best practices for deployment

### Best practices

Immutable Infrastructure

Deployment as Code

Measurement

Test

Cost effective

- Easy to use ASG
- Easy to predict
- Support spot instance
- Support scheduled instance

EC2 Pricing Model Score

(normalized RI hours + normalized Savings Plans hours +  
normalized Spot hours) / (total normalized EC2 hours)

## ▶ Best practices for deployment

### Best practices

Immutable Infrastructure

Deployment as Code

Measurement

Test

Cost effective

- Easy to use ASG
- Easy to predict
- Support spot instance
- Support scheduled instance

EC2 Pricing Model Score **92%-97%**

(normalized RI hours + normalized Savings Plans hours +  
normalized Spot hours) / (total normalized EC2 hours)

▶ What is DevOps ?

# Testing best practices

(feat. Declarative Synthetic testing as Code)

## ▶ Best practices for Synthetic monitoring

### Synthetic Monitoring

#### Synthetic monitoring



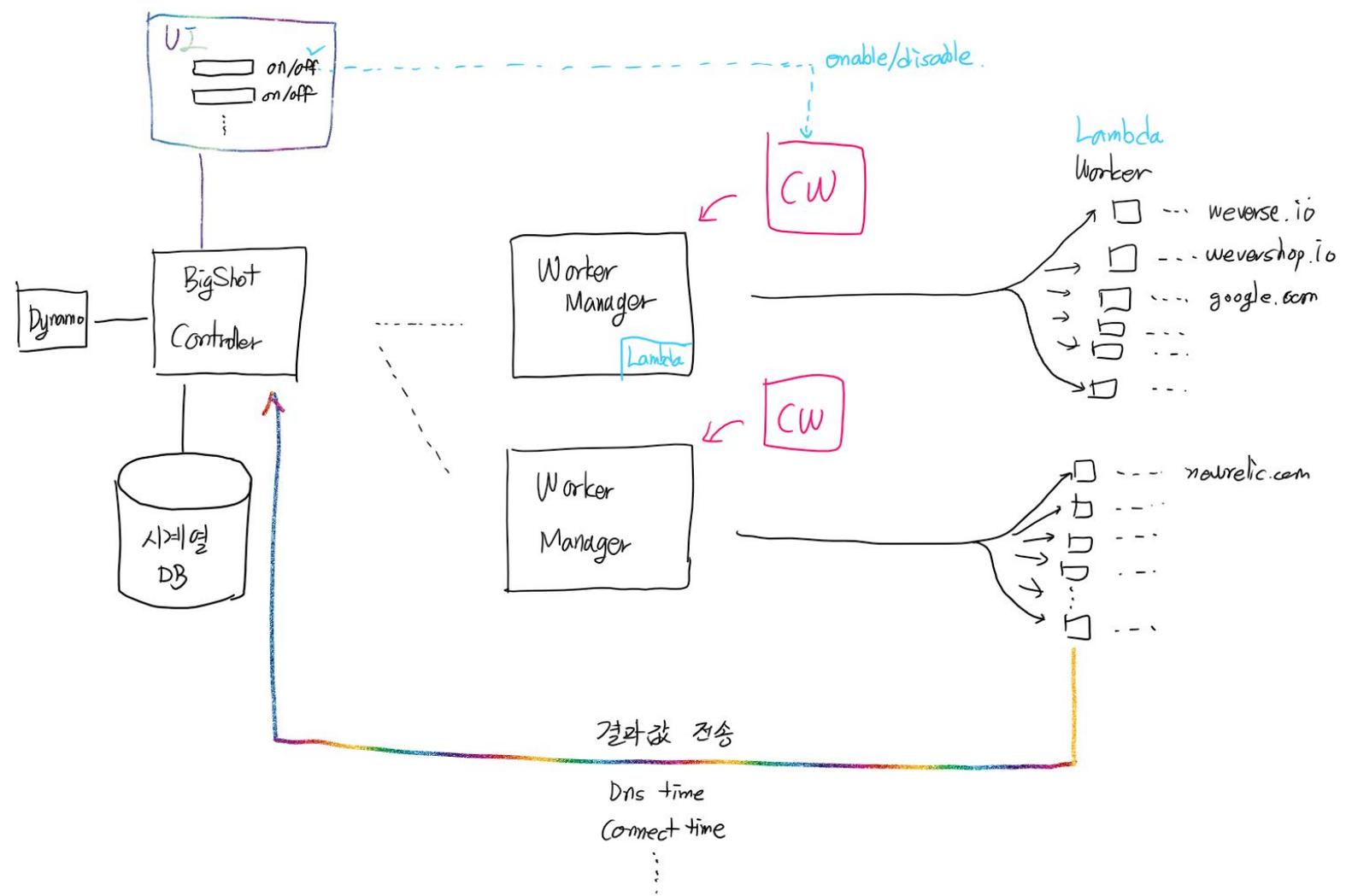
영어에서 번역됨 - 합성 모니터링은 트랜잭션의 에뮬레이션 또는 스크립팅 된 기록을 사용하여 수행되는 모니터링 기술입니다. 행동 스크립트는 고객이나 최종 사용자가 사이트, 응용 프로그램 또는 기타 소프트웨어에서 수행 할 동작이나 경로를 시뮬레이션하기 위해 만들어집니다. [위키백과\(영어\)](#)

- A way to check if application is down  
(Measure SLAs)
- A way to check a baseline for performance trends across regions
- A way to measure performance for application during peak and off traffic periods.
- A way to check if the problem caused by 3rd party service.
- A way to do “Real user monitoring”

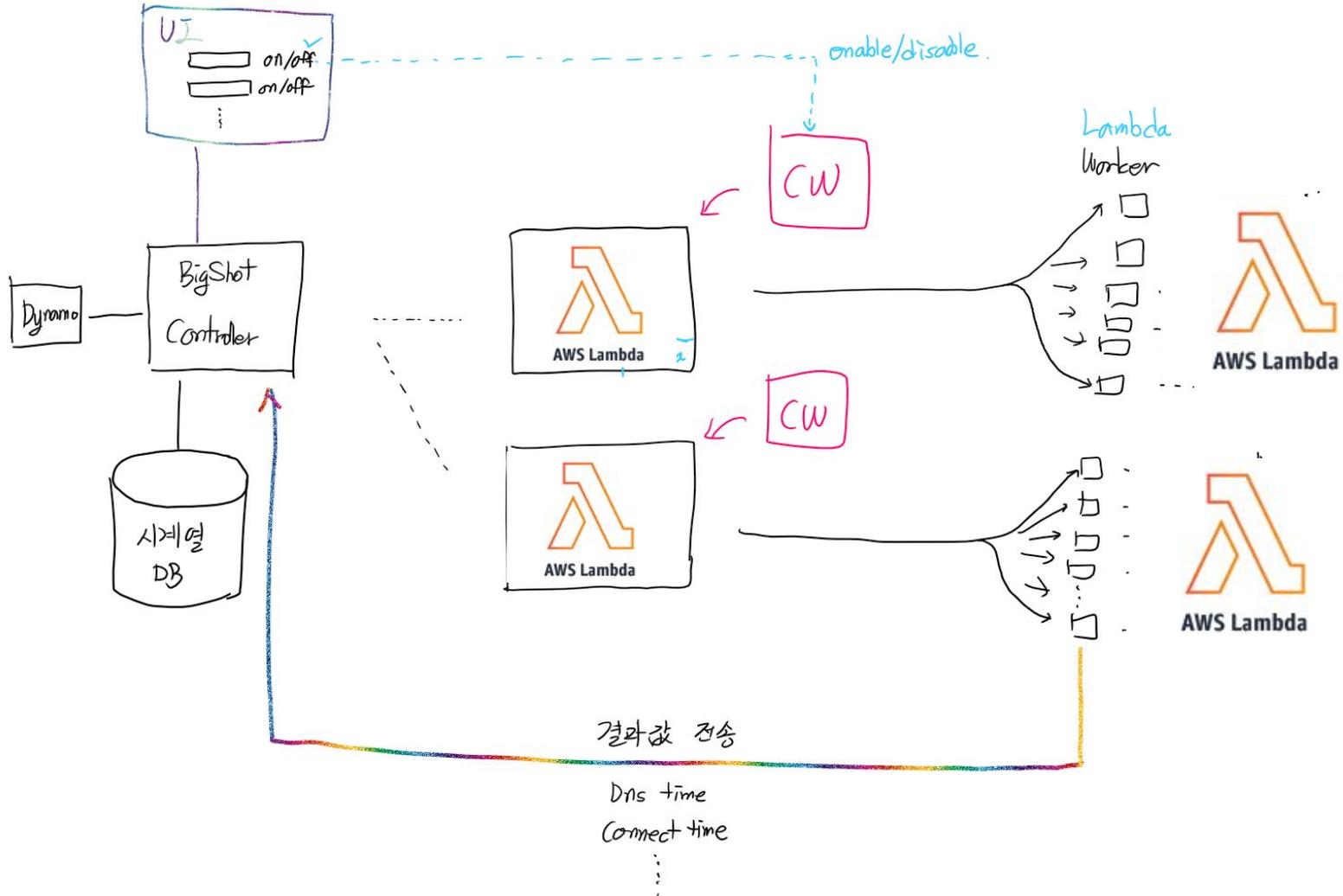
## ▶ Global service



# ▶ Bigshot architecture



# ▶ Bigshot architecture with Lambda – Cost effective



## ▶ Testing as Code



권수\_devops 6:53 PM

```
name: base-production
timeout: 300
interval: 300
slack_urls:
  - https://hooks.slack.com/services/XXXXXXXXX/31234141233/XXXXXXXXX
targets:
  - url: https://xxxxxxxx.io
    method: GET
  - url: https://api.xxxxxxxxx.io/api/v1/get/example
    method: GET
    header:
      authorization: Bearer XXXDDAJSKLDJQKWJDKASJDAKSDJKASJDKASJDAKSDJKAJSFKAJHSDQWOIEQQUIWR!@#!OKEDASDKSDJ
  - url: https://google.com
    method: GET

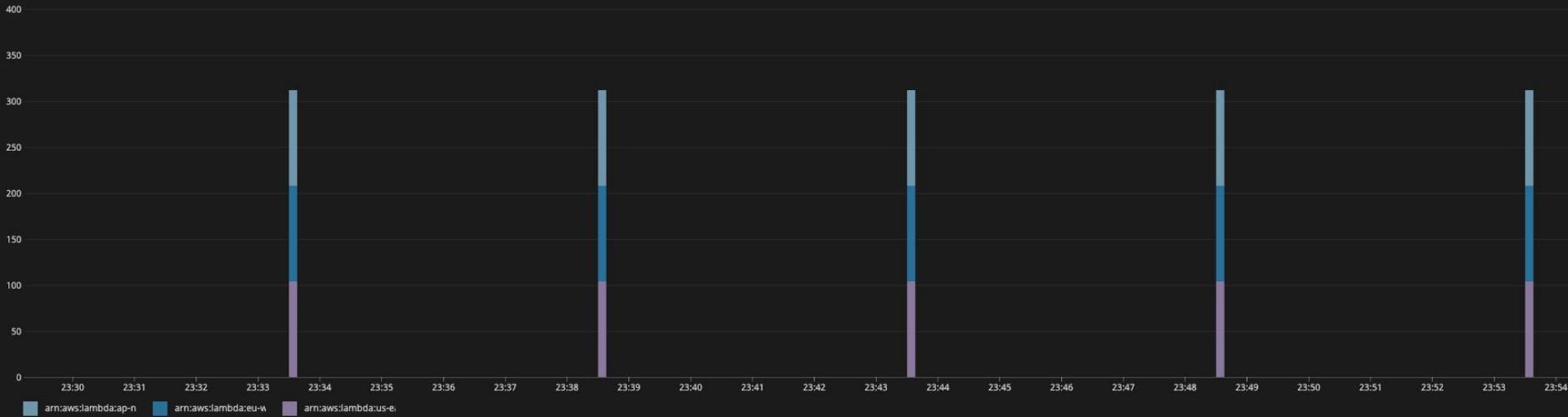
# Region configurations
regions:
  - region: ap-northeast-2
  - region: eu-west-1
  - region: us-east-1
```



주영S\_devops 6:53 PM

Testing as Code !! 너무 간단하고 보기 좋네요. (edited)

## ▶ Best practices



- Testing as Code
- Multiple regions
- Cost effective

▶ Project DevOps Art

# Project DevOps Art

함께 오픈소스 프로젝트를 만들어가실 분은 연락주세요

(junitarcana47@gmail.com)