DevOps Art Project

Terraform and Goployer
DevOps engineer at beNX & AWS Container hero

Juyoung Song, DevOps Engineer at beNX
Seoul, Korea

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.

Learn About Juyoung

Search Google!

Personal Github: https://github.com/jupitersong
Project Github: https://github.com/DevopsArtFactory
LinkedIn: https://www.linkedin.com/in/jupitersong
AWS Hero: https://aws.amazon.com/developer/community/heroes/juyoung-song
E-mail: jupitersong47@gmail.com

beNX is a subsidiary of Bighit Entertainment
Introduction

- DevOps Art Project
- Infrastructure as Code: Terraform
- Goployer
What is DevOps Art Project?
What is the project DevOps Art and purpose?

DevOps Art

DevOps 철학의 올바른 개념적 이해와 철학에 기반한 이상적인 구현을 위한 프로젝트

DevOps의 목적인 업무 속도와 효율화를 위해 다양한 코드를 공유하고 오픈소스를 개발하고 있습니다.

- Sharing Infrastructure code for best practices
- Opensource with Terraform
- CLI for automation
- Deployment tool
- Online workshop

Github: https://github.com/DevopsArtFactory
Terraform

Infrastructure as Code
Infrastructure as Code
코드로써의 인프라

Terraform is a tool for building, changing, and versioning infrastructure safely and efficiently.
테라폼은 인프라를 만들고, 변경하고, 기록하는 IaC를 위해 만들어진 도구로써, 문법이 쉬워 비교적 다루기 쉽고 사용자가 매우 많아 참고할 수 있는 예제가 많다.

IaC is a tool for building, changing, and versioning infrastructure safely and efficiently.
IaC는 코드로써의 장점, 즉 작성용이성, 재사용성, 유지보수 등의 장점을 가진다. 즉 빠르게 구축과 변경이 가능하여 신뢰할 수 있는 시스템을 만들어내기 위한 기술.
Past: Do you know Terraform and Packer??

Current: Do you use Terraform and Packer well??
DevOps art project : aws-provisioning

aws-provisioning project

Best practices for Terraform

- AWS
- variables
- symbolic link
- module
- workspace
- remote state
- convention

Workshop

- 이론 및 코드를 통한 실습

https://github.com/DevopsArtFactory/aws-provisioning
Goployer

A fast, powerful, simple deployment tool
Infrastructure as Code: Area and Cycle

- Source code
- Build
- Test
- Deployment
- Production
Infrastructure as Code: Area and Cycle

- Source code
- Build
- Test
- Deployment
- Production

Infrastructure as Code
Infrastructure as Code: Area and Cycle

- Source Repo
- Library manager
- Security Inspect
- Measurement
- C.I Tools
- Golden AMI
- Measurement
- Testing as code
- Synthetic automation
- Automate load test
- Measurement
- Deployment as code
- Immutable
- Measurement
- GitOps
- Monitoring as code
- Alerting as Code
- Chaos engineering
- Log management
- Analyzing
- Measurement
Infrastructure as Code: Area and Cycle

Source code:
- Source Repo
- Library manager
- Security Inspect
- Measurement

Build:
- C.I Tools
- Golden AMI
- Measurement

Test:
- Testing as code
- Synthetic automation
- Automate load test
- Measurement

Deployment:
- Deployment as code
- Immutable
- Measurement
- GitOps

Production:
- Monitoring as code
- Alerting as Code
- Chaos engineering
- Log management
- Analyzing
- Measurement
Opensource deployment tool Goployer: https://goployer.dev

AWS Deployment Tool

Goployer handles the whole processes of application deployment with AWS Autoscaling, Load Balancer, and EC2 Instances.

Production Level
Goployer provides most of features of deployment method with autoscaling group. You can use it in production environment.

Contributions welcome!
We make Pull Request contributions workflow on GitHub. New users are always welcome!

Booster your application
You don’t need to be worried about deployments. Just develop your application and deploy easily and comfortably.
DevOps philosophy with Goployer

Best practices for deployment

Best practices

Immutable Infrastructure

- 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

Deployment as Code

- Ensure each phase is the same

Measurement

- Create and use golden AMI by Packer

Cost effective

Simple
Immutable infrastructure

Best practices for deployment

Admin

- goployer deploy
- 1. create ASG
- 1. Notify Slack

GOPLOYER

new v002

exists v001
Immutable infrastructure

Best practices for deployment

1. Create instance
   goployer deploy
   GOPLOYER

2. Userdata boot up
   new v002

Admin

exists v001
Immutability infrastructure
Best practices for deployment

1. create ASG
2. userdata boot up
3. health check
3. Notify Slack

new v002
exists v001
Immutable infrastructure

Best practices for deployment

1. create ASG
2. userdata boot up
3. health check
4. Terminate exist ASG
4. Notify Slack

Blue/Green deployment!
Immutable infrastructure
Best practices for deployment

Admin → goployer deploy → GOPLOYER → 5. Notify Slack → current v002
DevOps philosophy with Goployer

Best practices for deployment

Best practices

Immutable Infrastructure

Deployment as Code

Measurement

Cost effective

Simple

service.yaml file

```
- project=hello
- repo=hello-deploy

stacks:
  - stack=ard
    polling_interval: 30s
    account: dev
    env: dev
    replacement_types: BlueGreen
    ssm_instance_profile: app-hello-profile
    ebs_optimized: true
    block_devices:
      - device_name: /dev/xvda
        volume_size: 15
        volume_type: gp2
        device_name: /dev/xvdb
        volume_type: st1
        volume_size: 500
    capacity:
      min: 1
      max: 2
    desired: 1
    autoscaling:
      autoscaling_policy
      alarms:
      lifecycle_callbacks:
      pre_terminate_instance:
        - service hello stop
    regions:
      - region: ap-northeast-2
        instance_type: t3.medium
        ssh_key: test-master-key
        ssh_id: ssh-rsa 4130345b5d2e6d4a
        use_public_subnet: true
        vpc: vpc-ard-apnortheast2
        detailed_monitoring_enabled: false
        security_groups:
          - hello-ard-apnortheast2
          - default-ard-apnortheast2
          - healthcheck_target_group: hello-ardapm2-ext
        availability zones:
          - ap-northeast-2a
          - ap-northeast-2b
          - ap-northeast-2c
    target_groups:
```
DevOps philosophy with Goployer
Best practices for deployment

Best practices

Immutable Infrastructure

Deployment as Code

Measurement

Cost effective

Simple

- If move it, measure it
- Get insight from everything
- metrics.yaml
- Dynamodb
  - Metrics for deployment: info, date
  - Metrics for server: uptime,
  - Stats: RequestCount,
## DevOps philosophy with Goployer

Best practices for deployment

<table>
<thead>
<tr>
<th>identifier</th>
<th>deployment_status</th>
<th>config</th>
<th>release_notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>hello-v</td>
<td>terminated</td>
<td>{&quot;manifest&quot;:&quot;deployments/hello.yml&quot;,&quot;manifest_s3_region&quot;:&quot;&quot;,&quot;ami&quot;:&quot;ami-0c89...&quot;}</td>
<td>By goployer</td>
</tr>
<tr>
<td>hello-v</td>
<td>deployed</td>
<td>{&quot;manifest&quot;:&quot;deployments/hello.yml&quot;,&quot;manifest_s3_region&quot;:&quot;&quot;,&quot;ami&quot;:&quot;ami-0c89...&quot;}</td>
<td>By goployer</td>
</tr>
<tr>
<td>hello-v</td>
<td>terminated</td>
<td>{&quot;manifest&quot;:&quot;deployments/hello.yml&quot;,&quot;manifest_s3_region&quot;:&quot;&quot;,&quot;ami&quot;:&quot;ami-0c89...&quot;}</td>
<td>By goployer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>terminated_date</th>
<th>statistics_record_time</th>
<th>start_date</th>
<th>deployed_date</th>
<th>uptime_hour</th>
<th>uptime_minute</th>
<th>uptime_second</th>
<th>stat</th>
</tr>
</thead>
<tbody>
<tr>
<td>2020-08-14T01:57:24Z</td>
<td>2020-08-14T01:57:24Z</td>
<td>2020-08-14T00:45:02Z</td>
<td>2020-08-14T00:47:36Z</td>
<td>1.163491</td>
<td>69.809457</td>
<td>4188.567437</td>
<td>{&quot;targetgroup...&quot;}</td>
</tr>
</tbody>
</table>

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 : 36676.025
2020-08-25T04:00:00Z Number : 32434.390554
2020-08-25T05:00:00Z Number : 46389.233862
total Number : 1027216.305707
Cost effective
Don’t waste money, Save money

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

Immutable Infrastructure

Deployment as Code

Measurement

Cost effective

Simple

EC2 Pricing Model Score
(normalized RI hours + normalized Savings Plans hours + normalized Spot hours) / (total normalized EC2 hours)
Best practices

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

Cost effective

Don’t waste money, Save money

EC2 Pricing Model Score is **96%**

(normalized RI hours + normalized Savings Plans hours + normalized Spot hours) / (total normalized EC2 hours)
DevOps philosophy with Goployer
Best practices for deployment

**Best practices**

- Developed by golang (No install)
- Simple commands
- Various powerful commands

**Immutable Infrastructure**

**Deployment as Code**

**Measurement**

**Cost effective**

Simple

goployer deploy --manifest=manifests/hello.yaml --stack=yourstack --region=ap-northeast-2
Goployer DEMO

Fast, Powerful, Simple deployment tool
Conclusion
DevOps Art Project

DevOps Art project

DevOps 철학의 올바른 개념적 이해와 철학에 기반한 이상적인 구현을 위한 프로젝트

Be Artist From Technician