

CLOUDZ LABS

Spinnaker를 활용한 배포 파이프라인 구성

2018.08.29



Table of Contents

01 | Spinnaker

02 | Hands-on

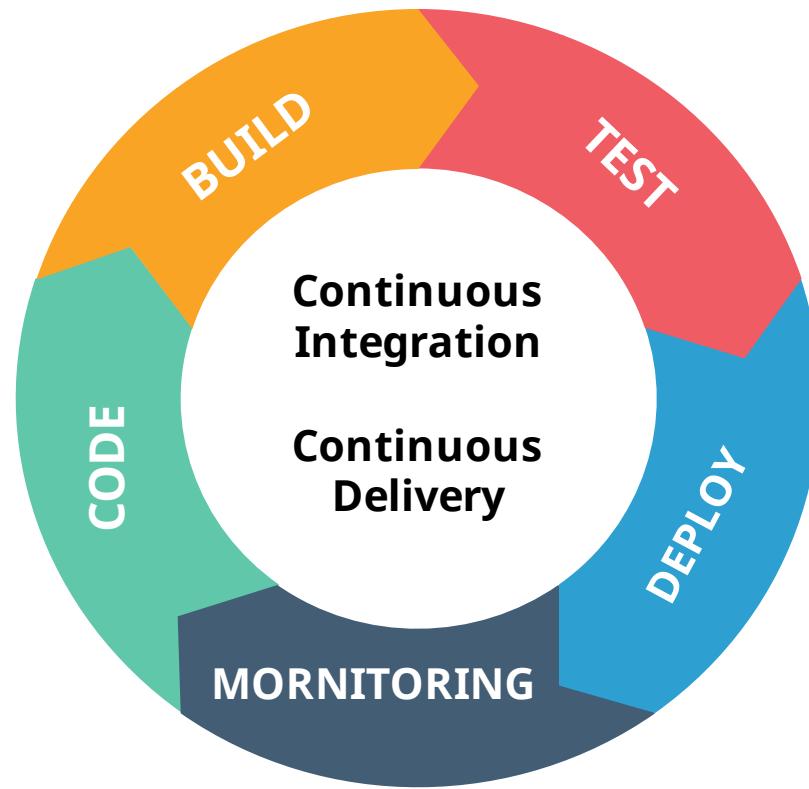
Spinnaker



Why Spinnaker?

Continuous Integration, Continuous Delivery

빠르고 지속적으로 질 좋은 시스템을 제공하기 위해 자동화된 Build, Test, Deploy를 파이프라인으로 구성하여 애플리케이션의 **지속적 통합, 지속적 배포**가 가능해야 함



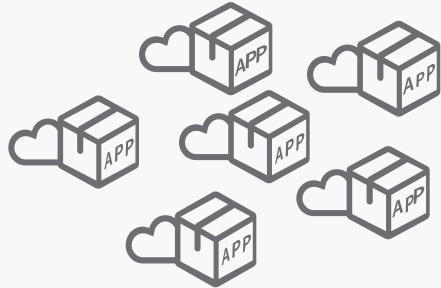
Why Spinnaker?



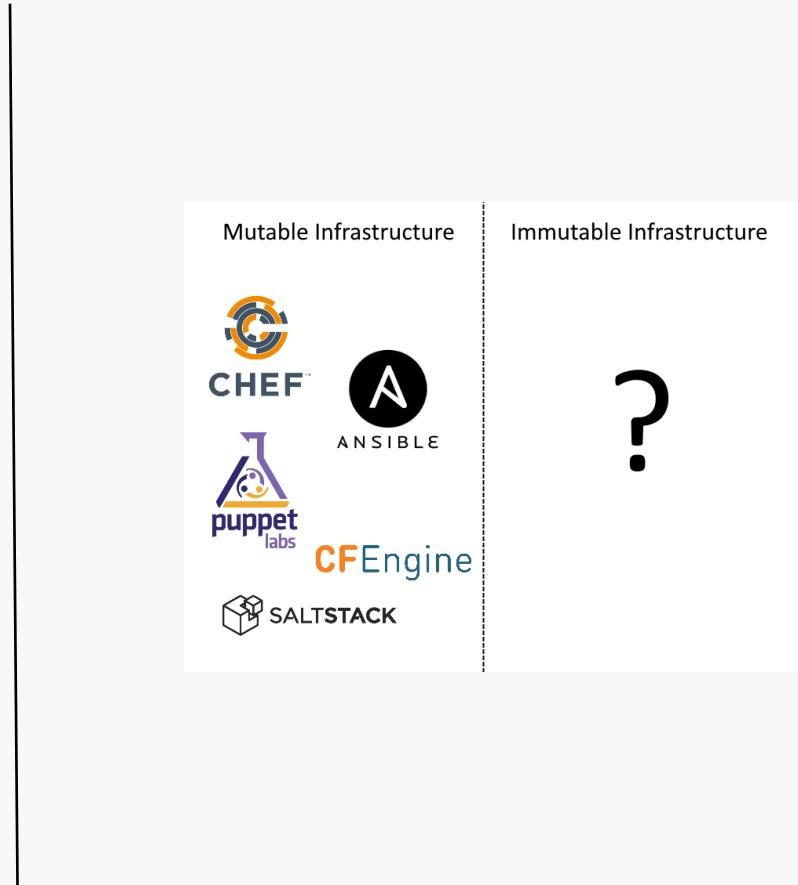
Monolithic



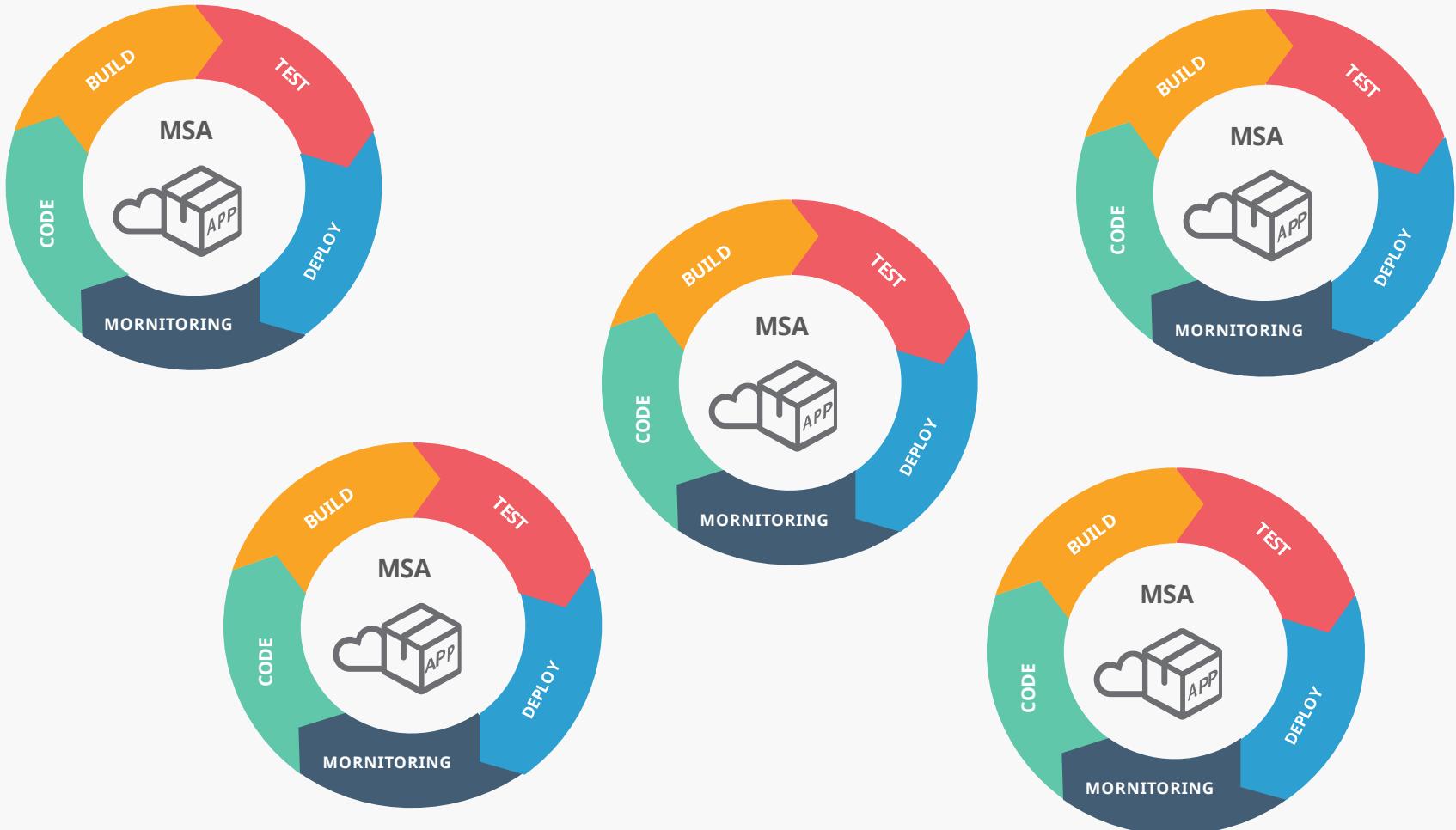
Microservice
Architecture



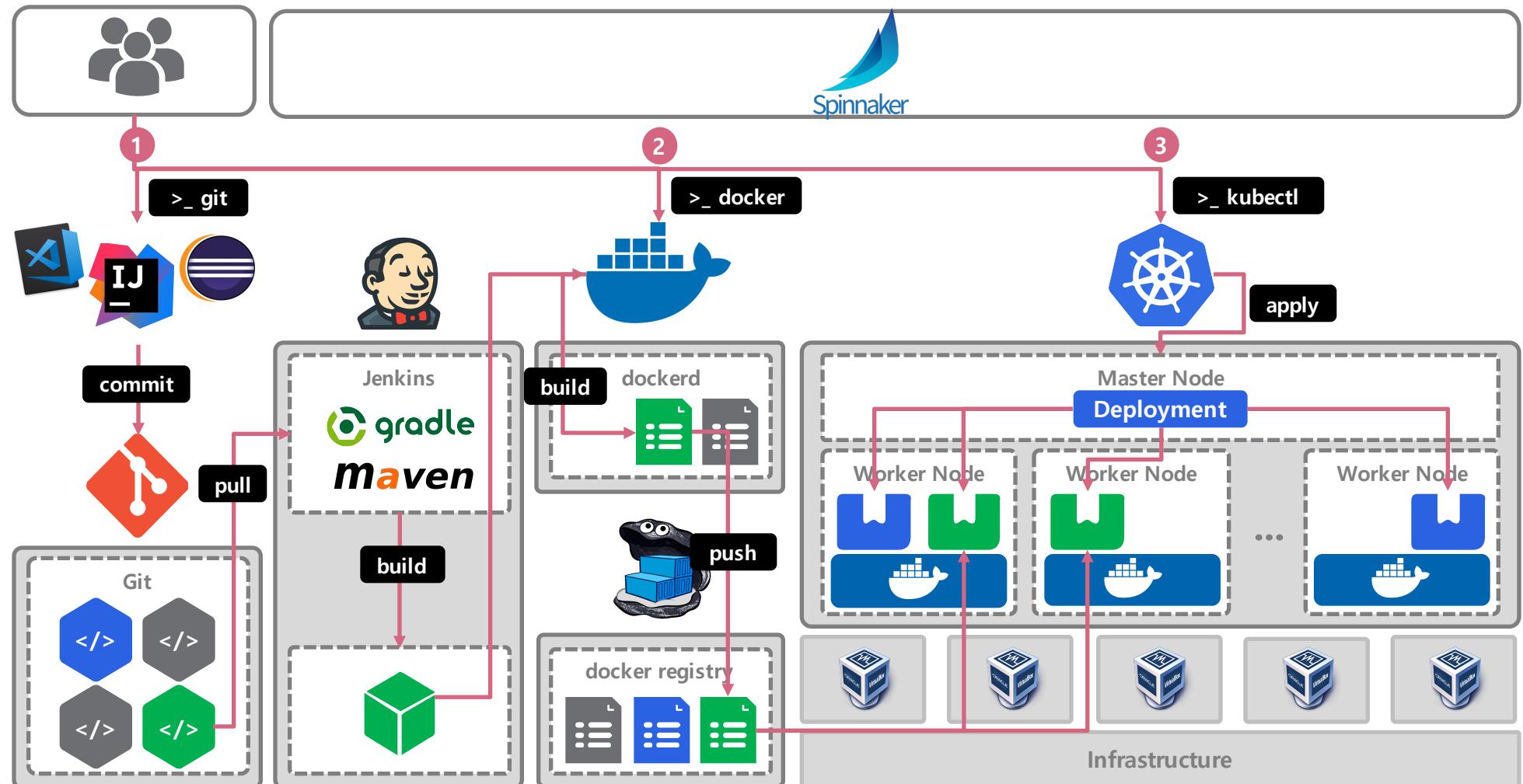
Cloud Application



Why Spinnaker?



CI/CD using Spinnaker



Why Spinnaker?

The Cloud Native Trail Map diagram illustrates the journey through various cloud native technologies. The path starts with Containerization, moves through Orchestration, Service Mesh, and Distributed Database, then branches into Container Runtime (with options like containerd, rkt, and CoreOS), Networking, Messaging, and Software Distribution. A dashed yellow circle highlights the 'CUCD' (Continuous Integration/Continuous Delivery) section, which includes Dockerfile CI/CD, Jenkins, and CircleCI. The diagram also features a 'HELP ALONG THE WAY' sidebar with sections for Training and Certification, Consulting Help, and Joining the CNCF community.

1. CONTAINERIZATION

- Normally done with Docker containers
- Any static application and dependencies (even Java!) written in an immutable way can be containerized
- Over time, you should move towards splitting static applications and moving future functionality as microservices

2. CUCD

- Setup Continuous Integration/Continuous Delivery (CI/CD) via Mat changes to your source code automatically results in a new container being built, tested and deployed to staging and eventually perhaps to production
- Setup automated rollouts, roll backs and testing

3. ORCHESTRATION

- Pick an orchestration solution
- Kubernetes is the market leader and you should select a Certified Kubernetes Platform or Distribution
- [https://kubernetes.io/](#)

4. OBSERVABILITY & ANALYSIS

- Pick solutions for monitoring, logging or tracing
- Docker CNCF projects Prometheus, Grafana, Stackdriver, Jaeger
- For tracing look at OpenTracing-compatible implementation like Jaeger

5. SERVICE MESH

- Connects services together and provides ingress from the Internet
- Service discovery, health-checking, routing, load balancing
- Consul, Envoy, Linkerd and Istio

6. NETWORKING

- To enable more flexible networking, use a CNCF-compliant network project like Calico, Flannel or Weave Net

7. DISTRIBUTED DATABASE

- When you need more resilience and scalability than a single database, this is a good option for running MySQL at scale through sharding

8. MESSAGING

- When you need higher performance than JSON REST, consider using gRPC

9. CONTAINER RUNTIME

- You can use alternative container runtimes, the most common, all of which are OCI compliant, are containerd, rkt and CoreOS

10. SOFTWARE DISTRIBUTION

- If you need to do secure software distribution, investigate Hoxton, an implementation of the Update Framework

HELP ALONG THE WAY

A. Training and Certification

Consider training offerings from CNCF and then take the exam to become a Certified Kubernetes Administrator [https://www.cncf.io/training](#)

B. Consulting Help

If you want assistance with Kubernetes and the surrounding ecosystem, consider leveraging a Kubernetes Certified Service Provider [https://cncf.io/csp](#)

C. Join CNCF's End User Community

For companies that don't offer cloud native services externally [https://cncf.io/euc](#)

WHAT IS CLOUD NATIVE?

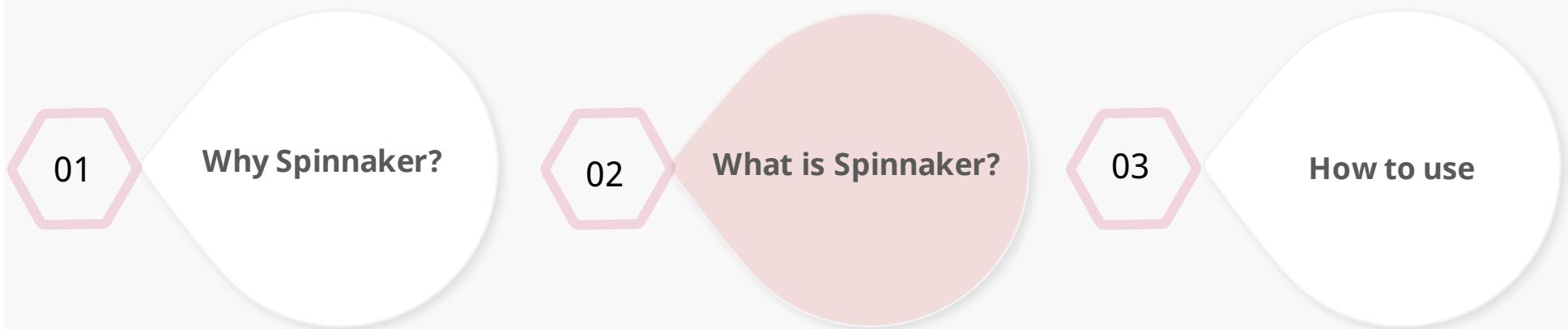
- **Openness:** Expose control of application system lifecycle
- **Observability:** Provide meaningful signals for observing state, health, and performance
- **Hatifiety:** Draw and shrink to fit in available resources and to meet fluctuating demand
- **Resiliency:** Fast automatic recovery from failures
- **Agility:** Fast deployment, iteration, and reconfiguration

[www.cncf.io](#) [info@cncf.io](#) v7.0

Setup Continuous Integrating/Continuous Delivery

Setup automated rollouts, roll backs and testing

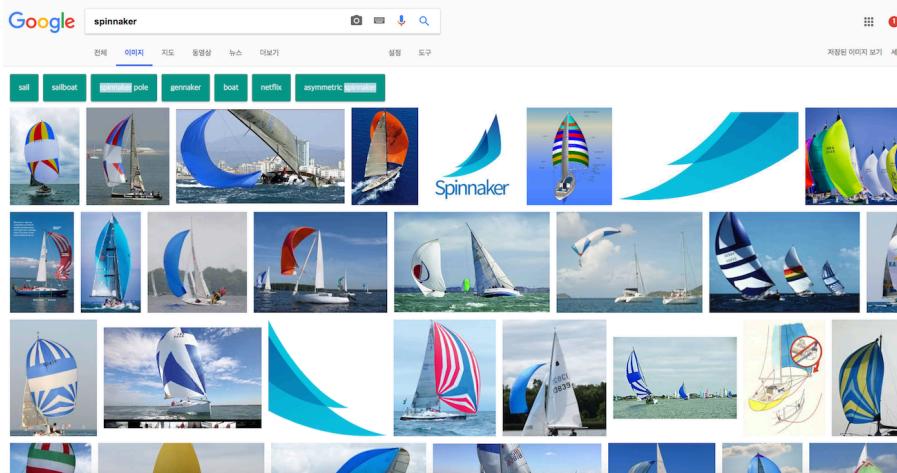
Spinnaker



What is Spinnaker?

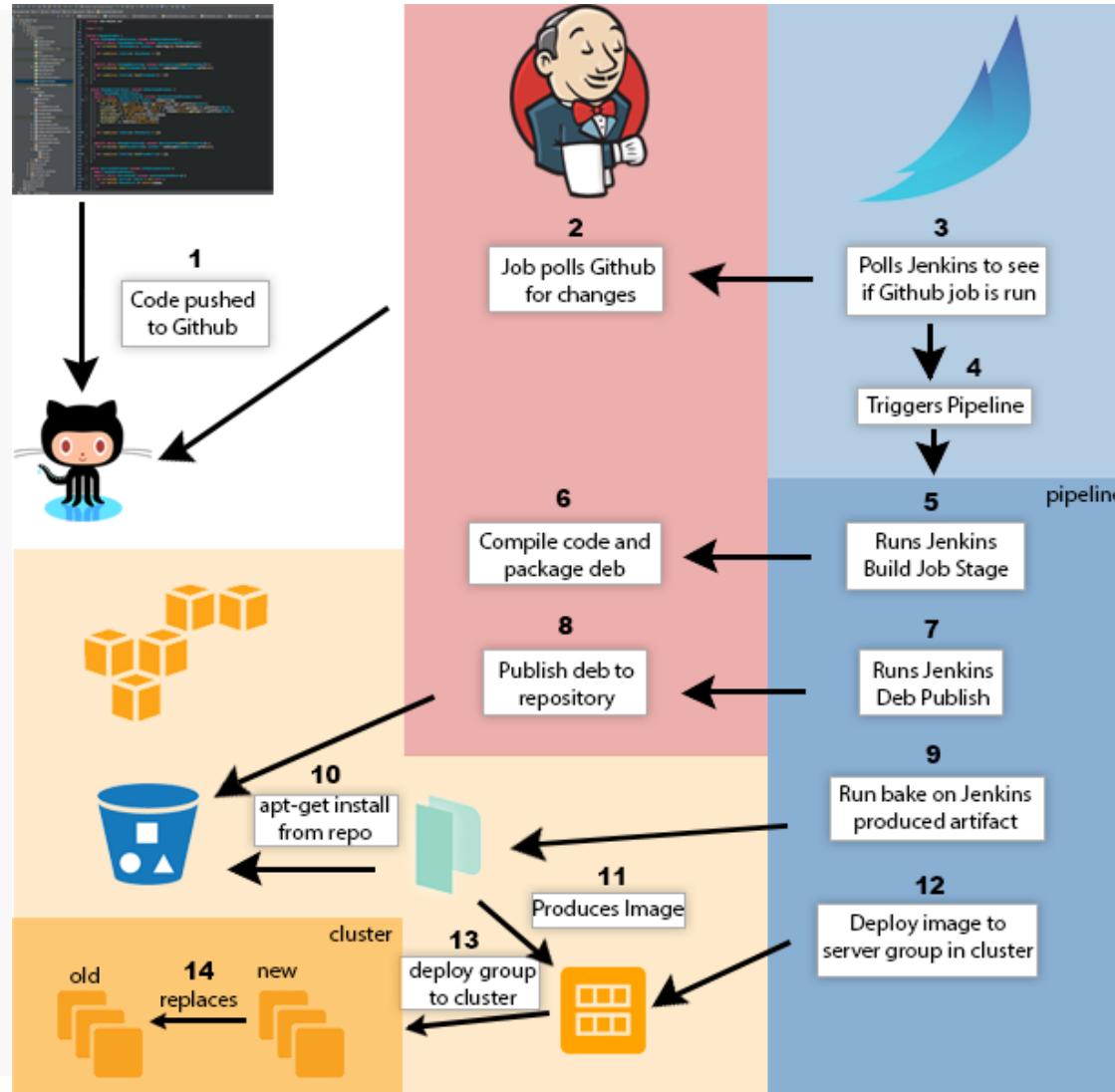
Spinnaker의 정의

영어 사전> 경주용 요트에 추가로 다는 큰 돛



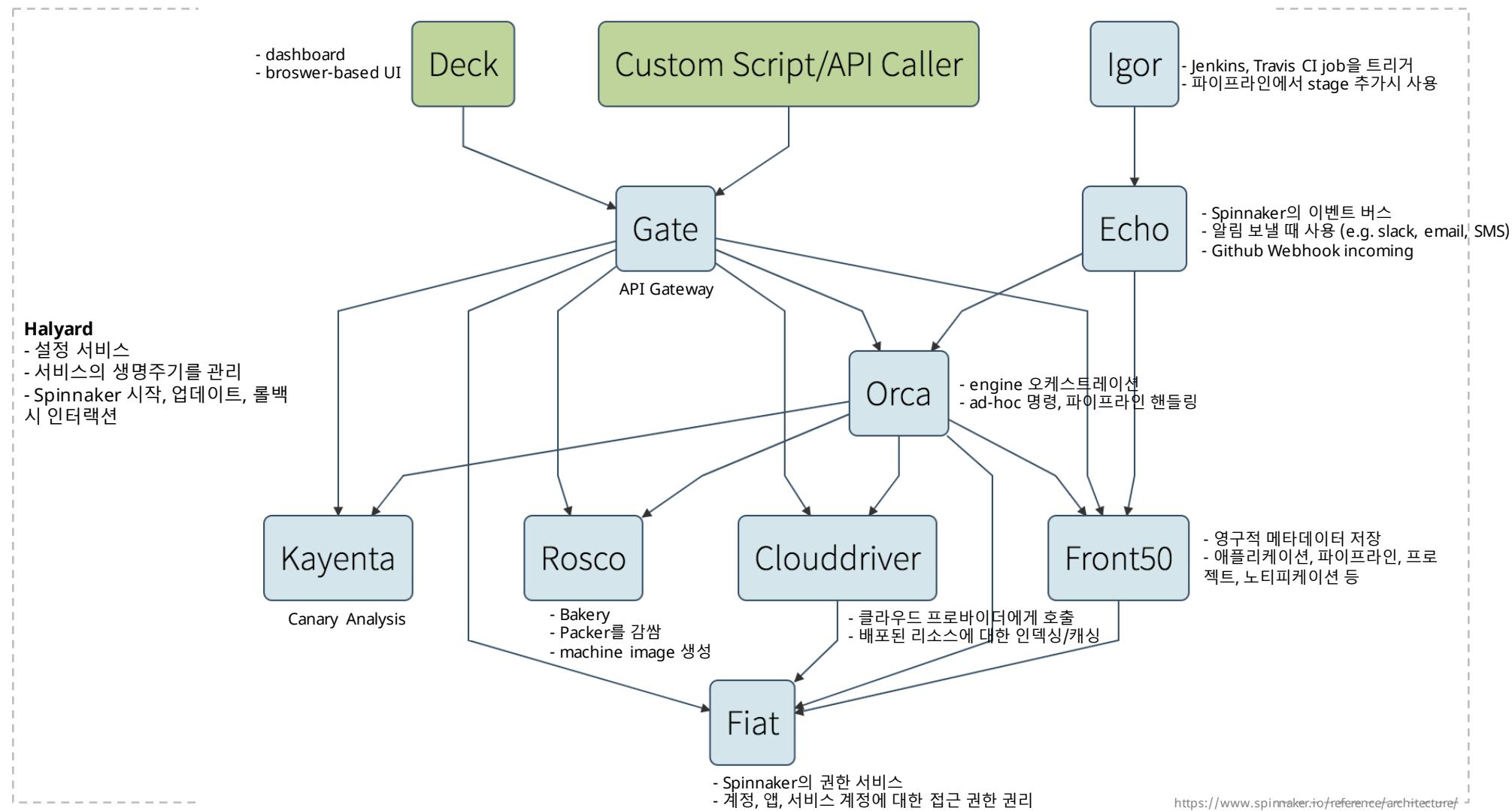
continuous delivery를 위한
배포 자동화 관리 솔루션

What is Spinnaker?



What is Spinnaker?

Spinnaker Architecture



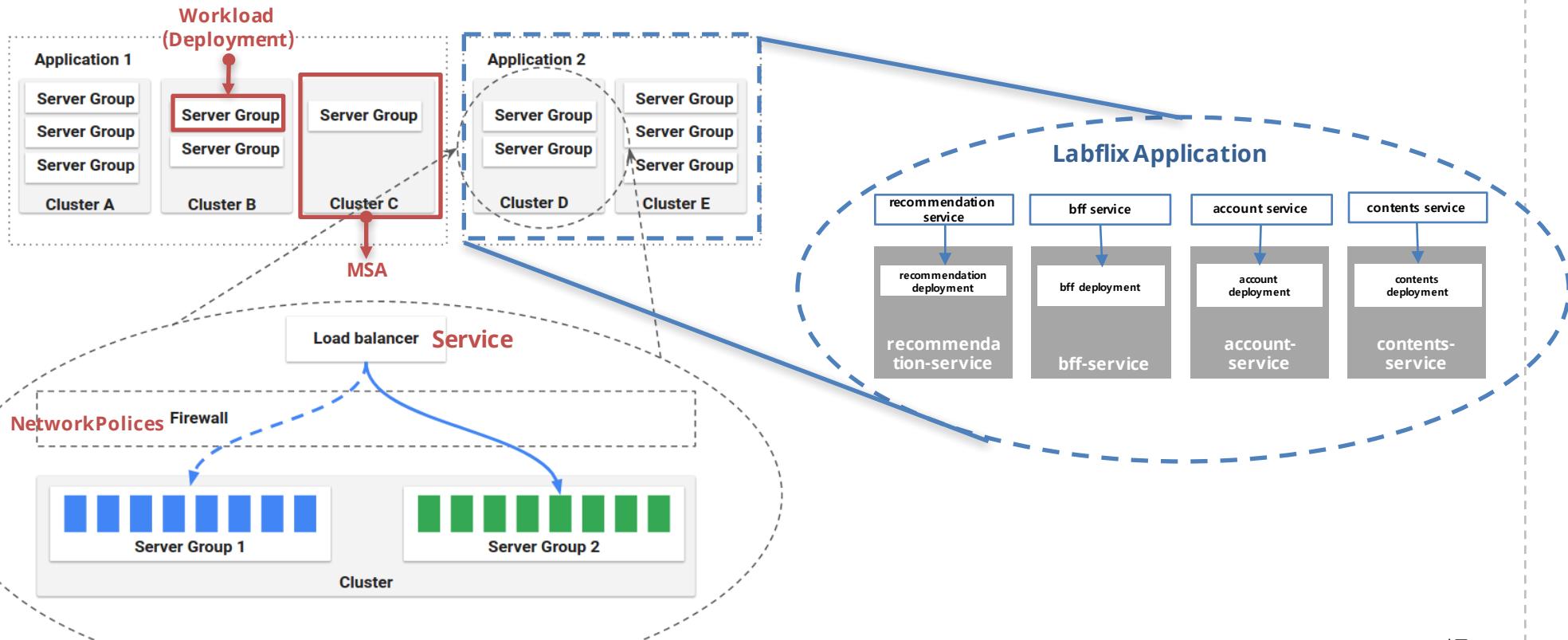
<https://www.spinnaker.io/reference/architecture/>

What is Spinnaker?

Spinnaker Concept - 1 Application Management

1 Application Management 클라우드에 있는 리소스를 조회, 관리

Server Group < Cluster < Application



*Kubernetes 기준

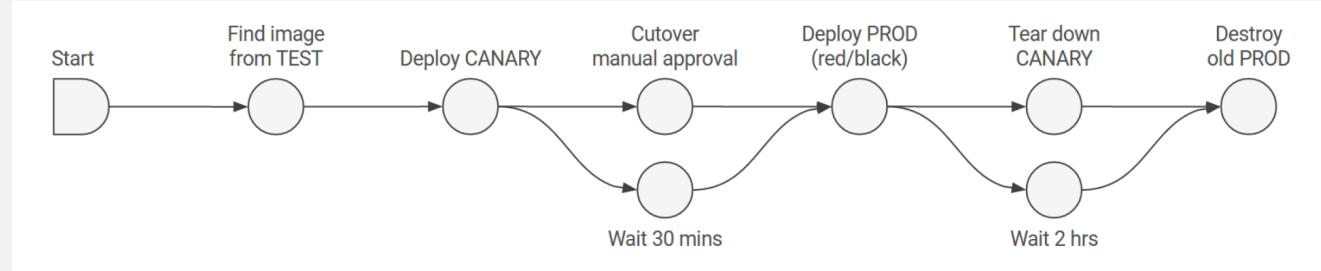
What is Spinnaker?

Spinnaker Concept - 2 Application Deployment

2 Application Deployment 지속적인 배포 워크 플로우 구성하여 배포를 자동화

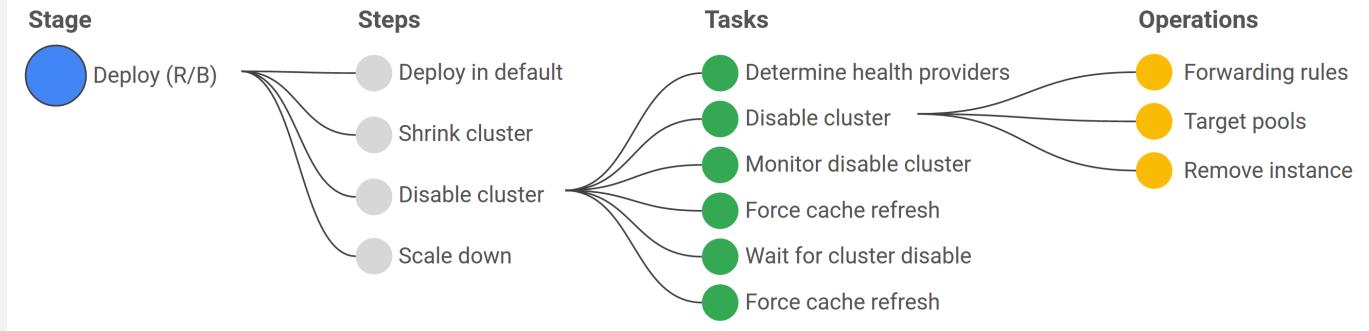
Pipeline

- Spinnaker의 배포 관리의 핵심
- 일련의 stage들로 구성
- Application을 생성
- 수동 또는 automated trigger로 시작



Stage

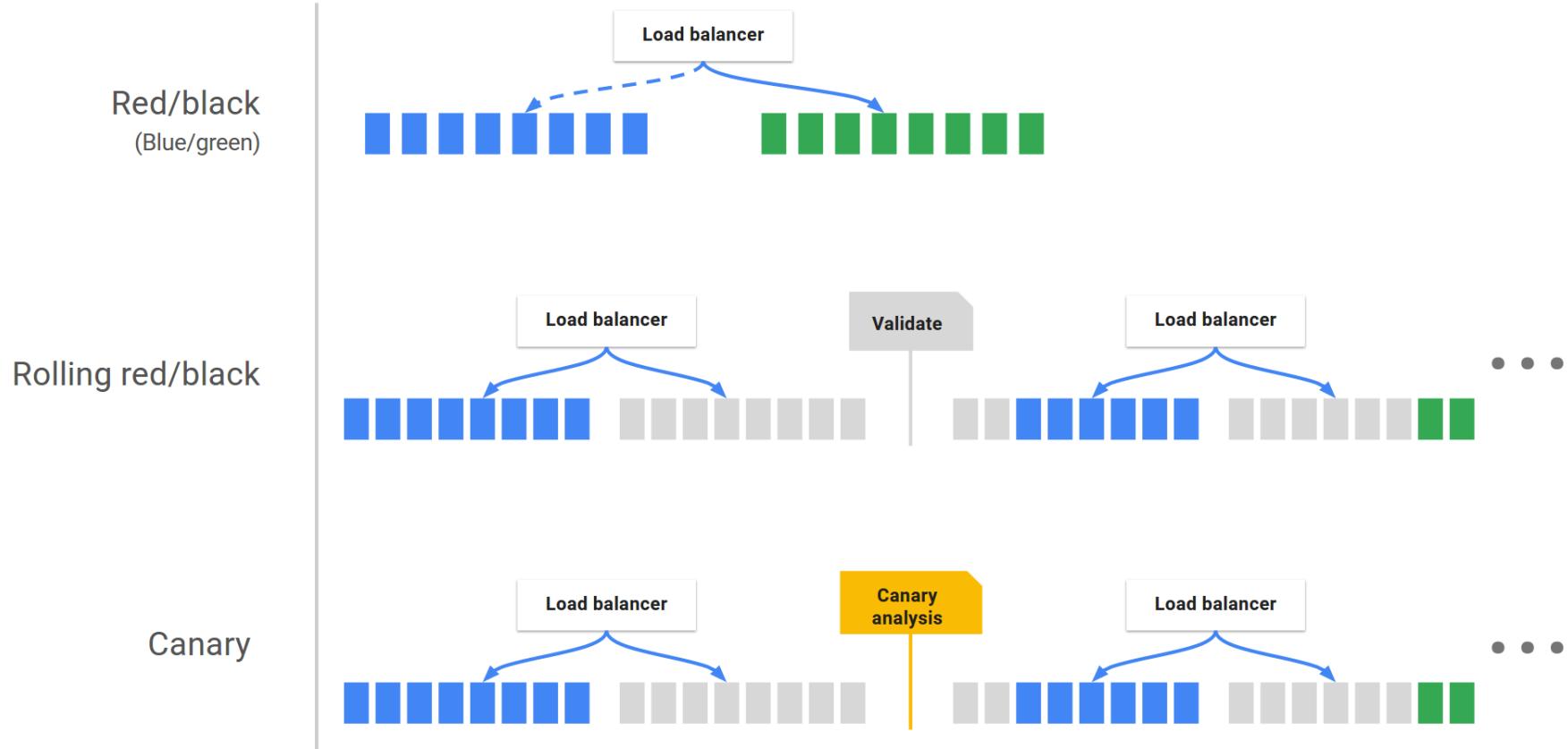
- Pipeline을 구성하는 가장 작은 단위
- 사용자가 수행하려는 동작을 설정



What is Spinnaker?

Spinnaker Concept - 2 Application Deployment

2 Application Deployment 다양한 배포 전략을 기본적으로 제공



Spinnaker



How to use

Spinnaker 구성요소 - Application

Application

- Spinnaker가 정의한 논리적 컨테이너 서비스의 집합
- 사용자가 직접 구성하여 사용

The screenshot shows the Spinnaker Applications interface. At the top, there is a navigation bar with links for SPINNAKER, Search, Projects, Applications (which is the active tab), and a search bar. To the right of the search bar are links for 'What's New' and 'Help'. Below the navigation bar is a header with the title 'Applications' and a search input field labeled 'Search applications'. On the far right of this header is a 'Actions' dropdown menu and a 'Create Application' button. The main content area is a table listing 14 applications. The columns are: Name, Created, Updated, Owner, Account(s), and Description. The 'Name' column lists the application names, which are all in blue and underlined, indicating they are links. The 'Owner' column shows the email address bckim0620@sk.com. The 'Account(s)' column for all entries shows 'my-k8s-v2-account'. The 'Description' column is empty for all entries.

| Name | Created | Updated | Owner | Account(s) | Description |
|---------------|-------------------------|-------------------------|------------------|-------------------|-------------|
| istio | - | - | | my-k8s-v2-account | |
| bong | 2018-08-16 01:54:03 PDT | 2018-08-16 01:54:04 PDT | bckim0620@sk.com | | |
| account | - | - | | my-k8s-v2-account | |
| alertmanager | - | - | | my-k8s-v2-account | |
| bff | - | - | | my-k8s-v2-account | |
| blackbox | - | - | | my-k8s-v2-account | |
| calico | - | - | | my-k8s-v2-account | |
| contents | - | - | | my-k8s-v2-account | |
| elasticsearch | - | - | | my-k8s-v2-account | |
| fluent | - | - | | my-k8s-v2-account | |
| fluentd | - | - | | my-k8s-v2-account | |
| grafana | - | - | | my-k8s-v2-account | |

How to use

Spinnaker 구성요소 - Pipeline

Pipeline

- Pipeline 목록, 실행 과정, 결과 및 자세한 정보를 볼 수 있음
- Pipeline 생성, 설정, 수동 실행

contents

PIPELINES

INFRASTRUCTURE

TASKS

CONFIG

bff-with-trigger

Trigger: enabled

WEBHOOK
5 days ago
index.docker.io/dtlabs/bff-service Version latest Status: SUCCEEDED Duration: 01:15

WEBHOOK
5 days ago
index.docker.io/dtlabs/bff-service Version 2.0 Status: SUCCEEDED Duration: 02:12

canary-with-istio (disabled) disabled

deploy all backend v1

delete all backend

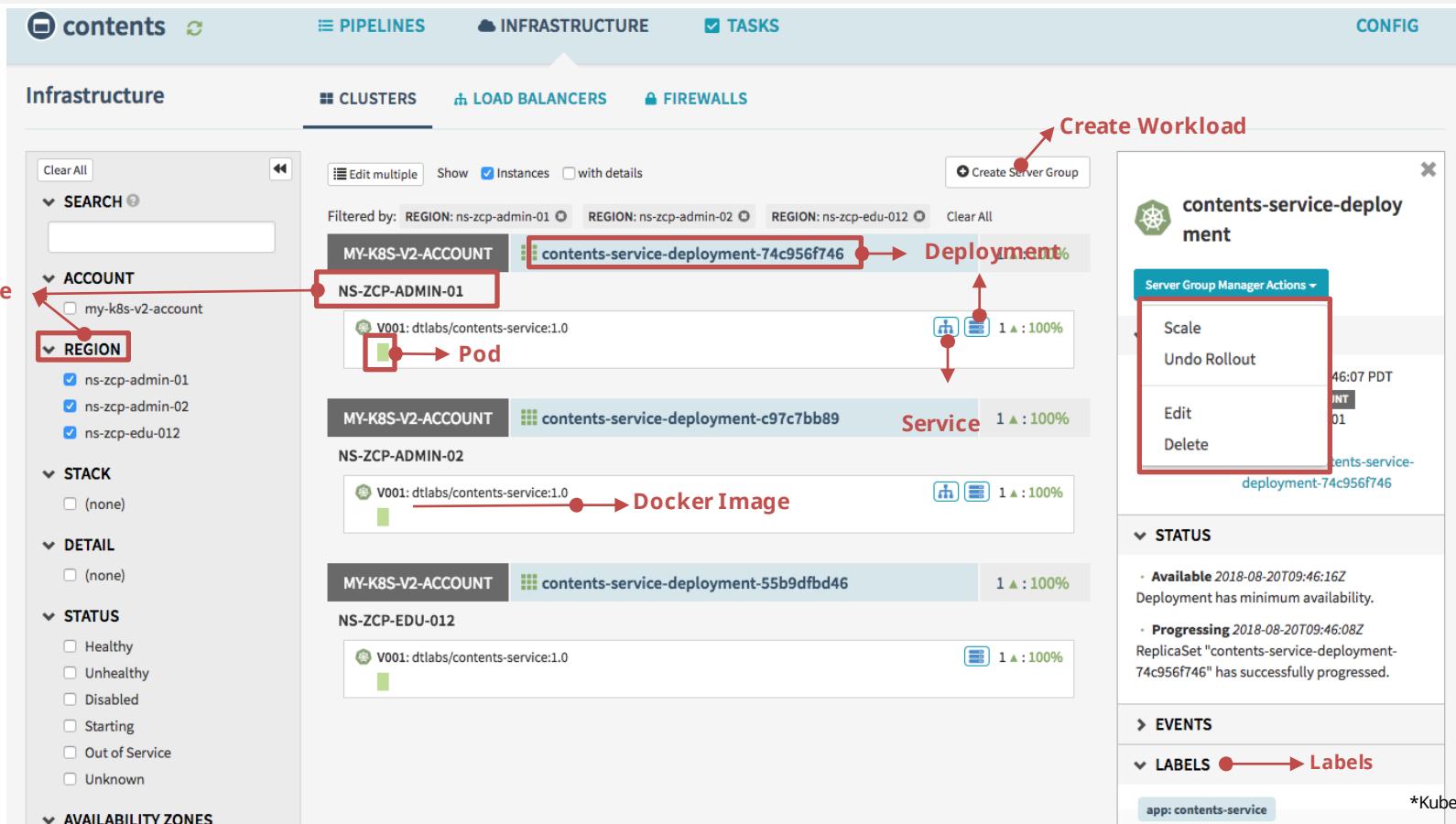
canary-maual-execution (disabled)

How to use

Spinnaker 구성요소 - Infrastructure

Cluster

- Application에 대한 인프라 정보를 조회할 수 있음
- Multi Cloud Provide의 Global한 배포 관리
- 실행 환경/배포 상태, 개별 인스턴스에 대한 메타데이터 조회
- Kubernetes의 Workload 정보 확인 및 설정 변경, 롤아웃 등을 UI로 수행할 수 있음



How to use

Spinnaker 구성요소 - Infrastructure

Load Balancers Kubernetes의 Service Controller

Firewall Kubernetes의 NetworkPolicy

The screenshot shows the Spinnaker Infrastructure page for the 'bff' application. The top navigation bar includes tabs for PIPELINES, INFRASTRUCTURE (selected), and TASKS, along with a CONFIG button. Below the navigation is a secondary navigation bar with tabs for CLUSTERS, LOAD BALANCERS (selected), and FIREWALLS.

The main content area displays a list of load balancers under the 'LOAD BALANCERS' tab. The list is organized by account and region. For each account, there is a summary card with the account name, the number of instances, and their current health status (e.g., 1 ▲ : 100% or 2 ▲ : 100%). Below each summary card is a list of individual instances, each represented by a green gear icon and a deployment name (replicaSet bff-service-deployment-...).

A search sidebar on the left provides filters for ACCOUNT (my-k8s-v2-account), REGION (ns-zcp-admin-02, ns-zcp-edu-01, ns-zcp-admin-01, ns-zcp-edu-02), STACK ((none) or service), DETAIL ((none)), and INSTANCE STATUS (Healthy). A 'Create Load Balancer' button is located at the top right of the main content area.

To the right of the main content, a detailed view of a specific deployment is shown in a modal window. The modal header is 'bff-service-deployment-v1-59c969d6fb'. It contains sections for INFORMATION (Created: 2018-08-21 21:31:59 PDT, Account: MY-K8S-V2-ACCOUNT, Namespace: ns-zcp-edu-01, Kind: replicaSet, Controller: Deployment bff-service-deployment-v1), EVENTS, and LABELS (app: bff-service, pod-template-hash: 1575258296).

How to use

Spinnaker 구성요소 - Task

Task 해당 Application에서 동작하고 있는 task를 한눈에 관리할 수 있는 뷰

contents PIPELINES INFRASTRUCTURE TASKS CONFIG

Task History

All Not Started Running Succeeded Terminal Canceled

Filter tasks by name, user, or task ID

| Task | Account | Region | Progress | Started | Ended | Running Time | User | Actions |
|--|---------|--------|---|-------------------------|-------------------------|--------------|-----------|---------|
| > Save pipeline 'bff-with-trigger' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-23 00:23:37 PDT | 2018-08-23 00:23:40 PDT | 00:03 | anonymous | |
| ▼ Save pipeline 'bff-with-trigger' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-21 22:57:39 PDT | 2018-08-21 22:57:42 PDT | 00:02 | anonymous | |
| ✓ Save Pipeline | | | | 2018-08-21 22:57:39 PDT | 2018-08-21 22:57:42 PDT | 00:02 | | |
| ✓ Wait For Pipeline Save | | | | 2018-08-21 22:57:42 PDT | 2018-08-21 22:57:42 PDT | 00:00 | | |
| Source Permalink | | | | | | | | |
| > Save pipeline 'bff-with-trigger' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-21 22:57:37 PDT | 2018-08-21 22:57:40 PDT | 00:02 | anonymous | |
| > Save pipeline 'canary-maul-excution' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-20 23:31:20 PDT | 2018-08-20 23:31:23 PDT | 00:03 | anonymous | |
| > Save pipeline 'Lㅁ' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-20 22:59:40 PDT | 2018-08-20 22:59:43 PDT | 00:03 | anonymous | |
| > Save pipeline 'delete all backend' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-20 01:26:46 PDT | 2018-08-20 01:26:49 PDT | 00:02 | anonymous | |
| > Save pipeline 'delete all backend' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-20 01:23:37 PDT | 2018-08-20 01:23:40 PDT | 00:02 | anonymous | |
| > Save pipeline 'delete all backend' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-20 01:22:08 PDT | 2018-08-20 01:22:11 PDT | 00:02 | anonymous | |
| > Save pipeline 'delete all backend' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-20 01:20:52 PDT | 2018-08-20 01:20:54 PDT | 00:02 | anonymous | |
| > Save pipeline 'delete all backend' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-20 01:18:58 PDT | 2018-08-20 01:19:00 PDT | 00:02 | anonymous | |
| > Save pipeline 'delete all backend' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-20 01:18:23 PDT | 2018-08-20 01:18:26 PDT | 00:03 | anonymous | |
| > Save pipeline 'delete all backend' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-20 01:14:19 PDT | 2018-08-20 01:14:22 PDT | 00:02 | anonymous | |
| > Save pipeline 'delete all backend' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-20 01:13:31 PDT | 2018-08-20 01:13:34 PDT | 00:02 | anonymous | |
| > Save pipeline 'delete all backend' | | | <div style="width: 100%;"><div style="width: 100%; background-color: #6aa84f;"></div></div> | 2018-08-20 01:12:59 PDT | 2018-08-20 01:13:02 PDT | 00:02 | anonymous | |

How to use

Spinnaker 구성요소 - Project

Project

사용자가 관리하려는 Cluster, Application, Pipeline의 집합

Configure Project

Project Attributes

- APPLICATIONS
- CLUSTERS
- PIPELINES

Project Name:

Owner Email: Enter an email address

Applications

Select...

Clusters

| Applications | Account | Stack | Detail |
|--------------|---------|-------|-------------|
| | | | Add Cluster |

Pipelines

(Add an application)

SPINNAKER Search Projects Applications

dna / Project Dashboard ▾

Project Configuration

Application Status

Filter by region / namespace ▾

| MY-K8S-V2-ACCOUNT | Last Push | ns-zcp-admin-01 | ns-zcp-admin-02 | ns-zcp-edu-01 | ns-zcp-edu-012 | ns-zcp-edu-02 | spinnaker |
|-------------------|-----------------|-----------------|-----------------|---------------|----------------|---------------|-----------|
| BFF | #0 20 hours ago | 1 ▲ : 100% | 1 ▲ : 100% | 2 ▲ : 100% | 1 ▲ : 100% | 2 ▲ : 100% | - |
| CONTENTS | #0 14 hours ago | 1 ▲ : 100% | 1 ▲ : 100% | - | 1 ▲ : 100% | - | 10 ▼ : 0% |
| RECOMMENDATION | #0 19 hours ago | 1 ▲ : 100% | 1 ▲ : 100% | - | 1 ▲ : 100% | - | - |

Pipeline Status

CONTENTS: bff-with-trigger (started 2018-08-22 03:52:55 PDT)

CONTENTS: canary-with-istio (started 2018-08-17 04:15:18 PDT)

How to use

Pipeline 생성

1. Configuration 설정

parameter, artifact, 알림 등 Pipeline 수행에 필요한 정보를 설정하는 필수 단계

The screenshot shows the Spinnaker interface for managing pipelines. At the top, there's a navigation bar with 'SPINNAKER', 'Search', 'Projects', and 'Applications'. Below that is a header with 'contents', 'PIPLINES' (selected), 'INFRASTRUCTURE', and 'TASKS'. The main content area displays a pipeline named 'bff-with-trigger'. A red box highlights the 'Configuration' stage in the pipeline graph. Below the graph, the 'Concurrent Executions' section is expanded, showing two checkboxes: one checked for 'Disable concurrent pipeline executions (only run one at a time)' and another unchecked for 'Do not automatically cancel pipelines waiting in queue.' The 'Automated Triggers' section is also partially visible.

Bff-with-trigger pipeline 설정 화면

설정 항목

AUTOMATED TRIGGERS 트리거링 설정

PARAMETERS 사용자로부터 입력 받을 파라미터 정보 설정

NOTIFICATIONS 파이프라인 시작, 완료, 실패 시 수신을 여부 설정함. email, Slack 등 지원

DESCRIPTION 파이프라인에 대해 적고 싶은 내용을 적는 곳

EXPECTED ARTIFACTS 해당 파이프라인이 실행될 때 산출될 것으로 기대되는 아티팩트 정의

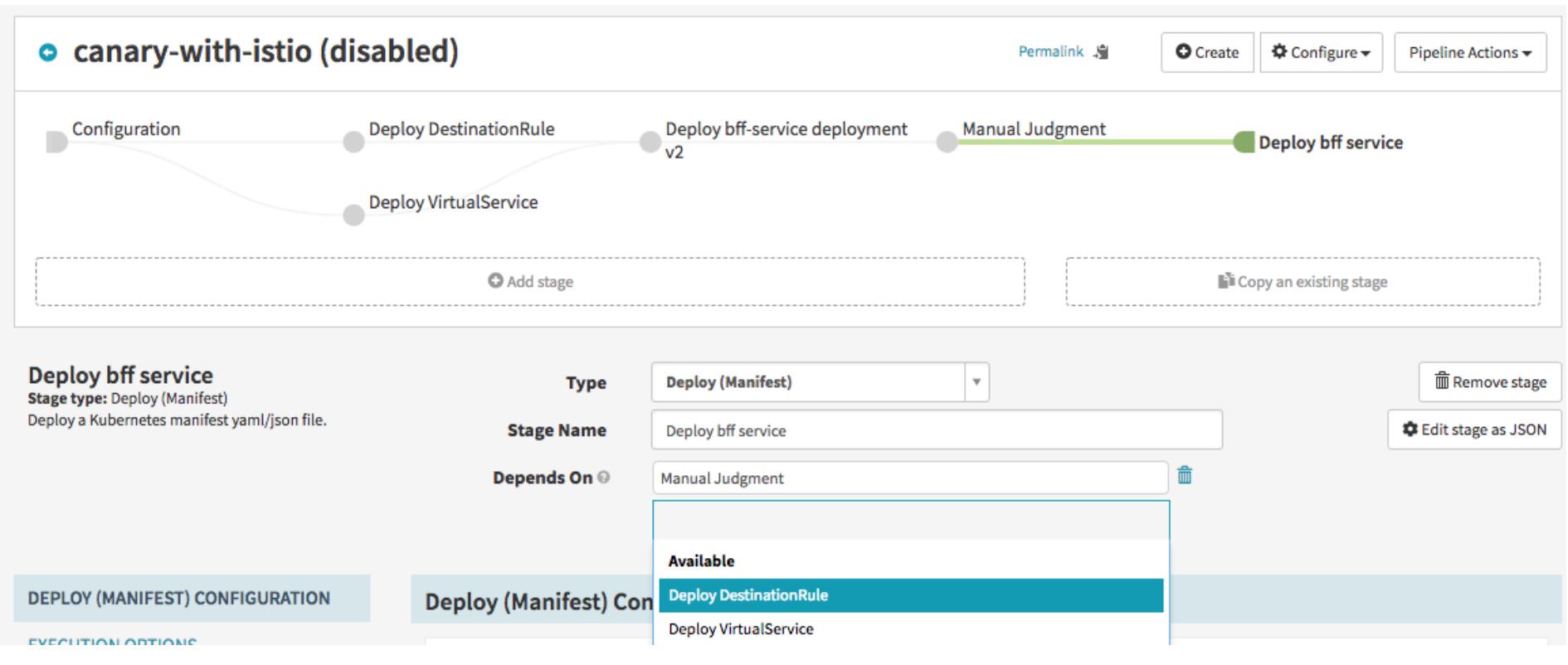
How to use

Pipeline 생성

2. Stage 설정

Stage 기본 설정

- Stage Type: 다른 stage copy 가능
- Pipeline명
- Stage 순서: Depends On 속성으로 설정



How to use

Pipeline 생성

Stage Type (K8S V2 기준)

Bake Image Baking. Helm 사용

Check Preconditions 클러스터 사이즈나 정규식을 이용한 상태 점검

Delete K8S 객체 삭제

Deploy 배포

Find Artifact from Execution 다른 파이프라인을 실행한 결과물 바인드

Find Artifact from Resource K8S 리소스를 찾아서 바인드

Manual Judgement 사용자의 승인

Jenkins Jenkins job 실행

Patch K8S의 객체를 Patch

Pipeline 지정한 파이프라인 실행

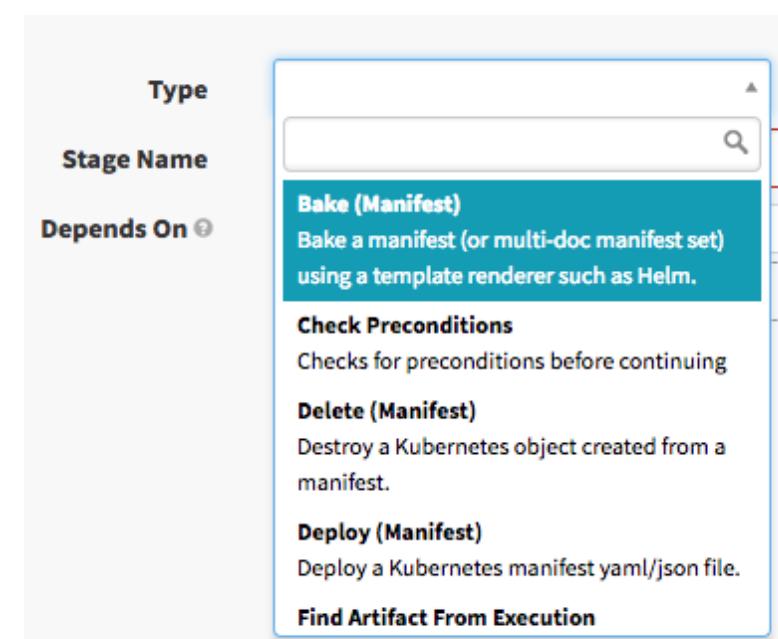
Scale Configuration K8S에 객체를 scaling

Script 스크립트 실행

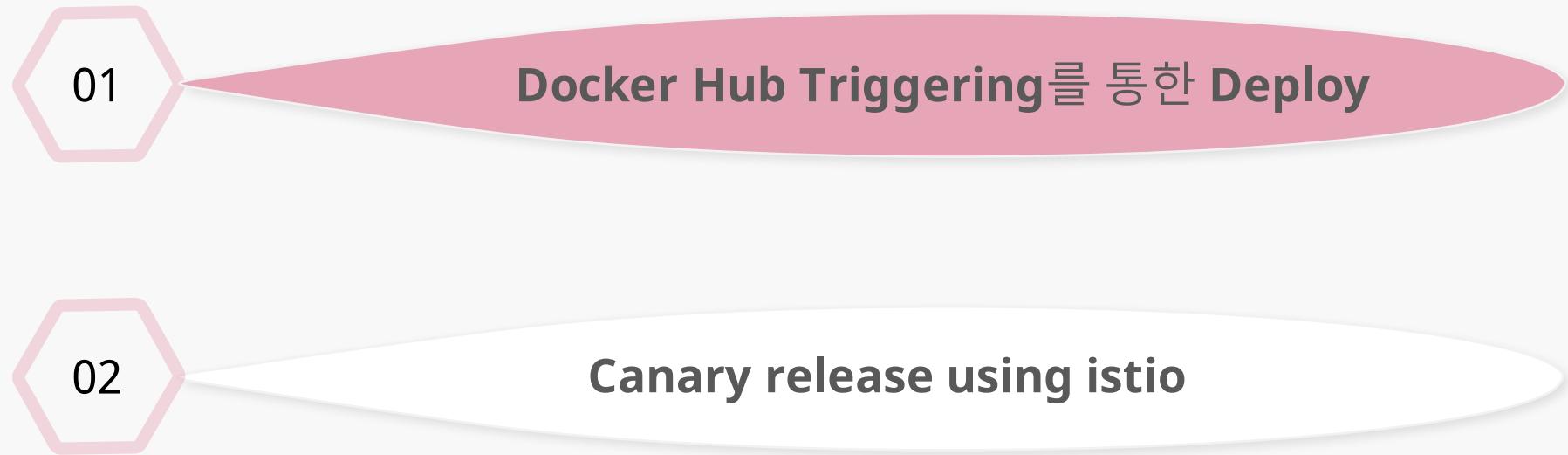
Undo Rollout K8S 객체의 상태를 원하는 revision number로 롤아웃

Wait 일정 시간 대기

Webhook HTTP의 GET, HEAD, POST, PUT, DELETE를 지정하여 Webhook 수행



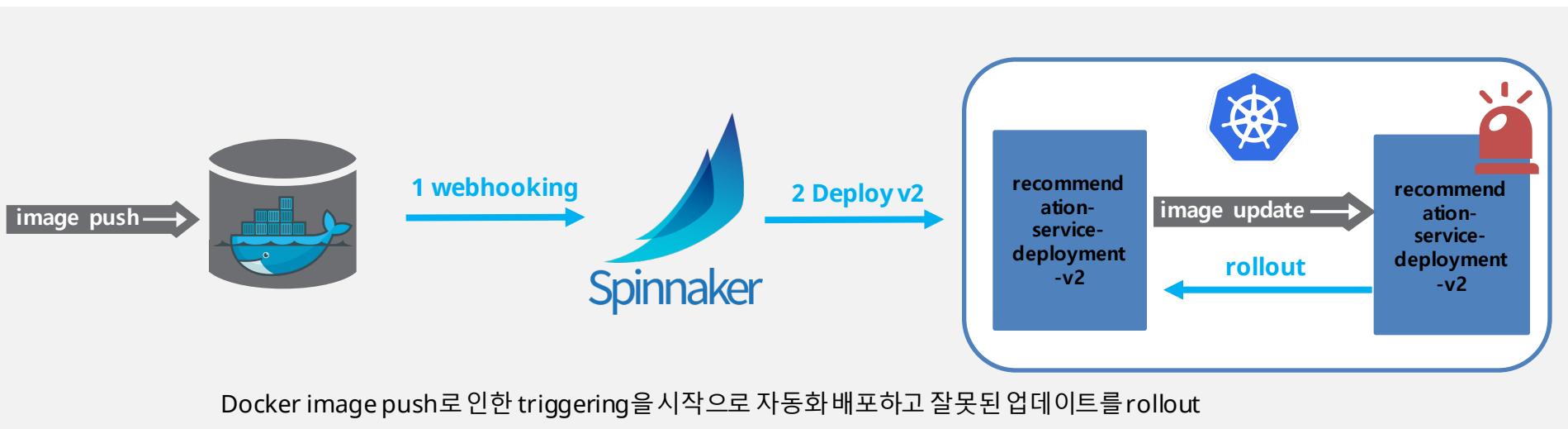
Hands-on: Deploy Pipeline 생성하기



Docker hub triggering

일반 배포 자동화

시나리오



Docker hub triggering

1.2 Application 생성

1 Spinnaker Dashboard에 접속 <http://spinnaker.zcp-dtlabs.jp-tok.containers.mybluemix.net/>

2 Application 생성

2-1 상단의 **Applications** 클릭 > 우측 **Actions** 버튼을 클릭하여 **Create Application** 선택



The screenshot shows the Spinnaker Dashboard interface. At the top, there's a navigation bar with 'SPINNAKER', 'Search', 'Projects', and 'Applications' tabs. The 'Applications' tab is selected and highlighted with a red arrow. On the right side of the dashboard, there's a 'Actions' dropdown menu with a 'Create Application' option, also highlighted with a red arrow. Below the navigation bar, there's a table titled 'Applications' with columns: Name, Created, Updated, Owner, Account(s), and Description.

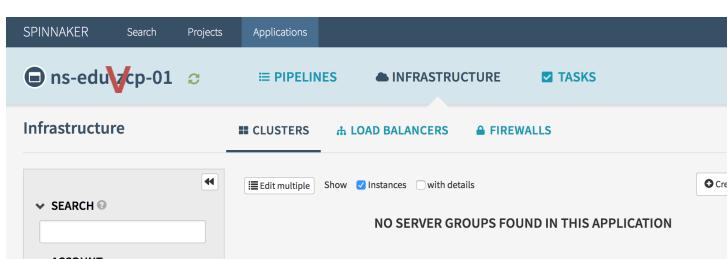
New Application

| | | |
|---|--|--------------------|
| Name * | <input type="text" value="Enter an application name"/> | Labsflix\${number} |
| Owner Email * | <input type="text" value="Enter an email address"/> | |
| Repo Type | Select Repo Type | |
| Description | <input type="text" value="Enter a description"/> | |
| Instance Health | <input type="checkbox"/> Consider only cloud provider health when executing tasks <input type="checkbox"/> Show health override option for each operation | |
| Instance Port | 80 | |
| Pipeline Behavior | <input type="checkbox"/> Enable restarting running pipelines | |
| <small>* Required</small> | | |
| <input type="button" value="Cancel"/> <input type="button" value="Create"/> | | |

2-2 Application 정보 입력

- Application명과 Email 주소 입력
- **Create** 버튼 클릭

✓ 생성된 Application 확인



The screenshot shows the Spinnaker Dashboard Infrastructure page. At the top, there are tabs for 'SPINNAKER', 'Search', 'Projects', 'Applications', 'PIPELINES', 'INFRASTRUCTURE', and 'TASKS'. The 'INFRASTRUCTURE' tab is selected. Below the tabs, there's a section for 'Infrastructure' with sub-sections for 'CLUSTERS', 'LOAD BALANCERS', and 'FIREWALLS'. A search bar is at the bottom left. The main area displays a table for 'SERVER GROUPS' with one entry: 'ns-eduVcp-01'. A note at the bottom right says 'NO SERVER GROUPS FOUND IN THIS APPLICATION'.

Docker hub triggering

3 Pipeline 생성

3-1 상단의 **PIPELINES** 클릭 – 하단의 Configure a new pipeline

ns-edu-zcp-01

No pipelines configured for this application.

[Configure a new pipeline](#)

3-2 Type: Pipeline 선택

Pipeline Name: 'Deploy recommendation v2' 입력
Create 버튼 클릭

Create New Pipeline

Type: Pipeline

Pipeline Name: Deploy recommendation v3

[Cancel](#) [Create](#)

✓ Pipeline 생성 확인

u-zcp-01

✓ Pipeline 생성 확인

Deploy recommendation v3

Configuration

Concurrent Executions

Automated Triggers

Docker hub triggering

4 Docker Image Webhook 적용

Docker Image hooking에 사용할 특정 URL을 만든다.

Configuration stages의 'Automated Triggers'에서 다음과 같이 설정

- **Type: Webhook** 선택
- **Source:** domain 뒤에 올 URL name 입력
dockerhub-recommendation 입력

Deploy recommendation v3

Configuration

CONCURRENT EXECUTIONS

AUTOMATED TRIGGERS (1)

PARAMETERS

NOTIFICATIONS

DESCRIPTION

EXPECTED ARTIFACTS

Automated Triggers

Type: Webhook Executes the pipeline when a webhook is received.
{ http://spinnaker-api.zcp-dtlabs.jp-tok.containers.mybluemix.net/webhooks/webhook/dockerhub-recommendation }

Source: dockerhub-recommendation

Payload Constraints

PUBLIC REPOSITORY

dtlabs/recommendation-service ☆

Last pushed: an hour ago

Webhooks

Workflows

TRIGGER EVENT

WEB HOOKS +

Image Pushed

When an image is pushed to this repo, your workflows will kick off based on your specified webhooks. [Learn More](#)

Spinnaker
{ http://spinnaker-api.zcp-dtlabs.jp-tok.containers.mybluemix.net/webhooks/webhook/dockerhub-recommendation }

Docker hub triggering

4 Parameter 설정

Namespace용 parameter를 생성한다.

Configuration stages의 Parameters에서 다음과 같이 설정

- **Name:** namespace
- **Default Value:** 배포하려는 namespace

| Parameters | | | | |
|----------------------------------|--|-------|--------------------------------------|--|
| | Name | Label | | |
| <input type="button" value="↑"/> | namespace | | <input type="button" value="trash"/> | |
| Required | <input checked="" type="checkbox"/> | | | |
| Description | <input type="text"/> | | | |
| Default Value | <input type="text" value="본인의 namespace"/> | | | |
| Show Options | <input type="checkbox"/> | | | |

Docker hub triggering

5 Kubernetes에 Deployment 배포

배포를 위한 파이프라인을 생성한다.

5-1 **Add stage** 클릭 - 다음과 같이 설정

- **Type: Deploy (Manifest)** 선택
- **Stage Name:** Stage명 입력
Deploy recommendation v2 입력

The screenshot shows a pipeline configuration interface with a green progress bar at the top. The bar has two segments: 'Configuration' (grey) and 'Deploy recommendation v2' (green). Below the bar, there is a button labeled '+ Add stage' with a red arrow pointing to it. The main configuration area is titled 'Deploy recommendation v3.0' and specifies 'Stage type: Deploy (Manifest)'. It includes fields for 'Type' (set to 'Deploy (Manifest)'), 'Stage Name' ('Deploy recommendation v2'), and 'Depends On' (a dropdown set to 'Select...'). A 'Create' button is visible in the top right corner.

Deploy recommendation v2

Configuration

+ Add stage

Deploy recommendation v2

Deploy recommendation v3.0

Stage type: Deploy (Manifest)

Deploy a Kubernetes manifest yaml/json file.

Type

Deploy (Manifest)

Stage Name

Depends On

Select...

Create

Docker hub triggering

5 Kubernetes에 Deployment 배포

배포할 Deployment의 매니페스트를 입력한다.

5-2 하단의 *Deploy (Manifest) Configuration* 설정

Basic Settings

Account * my-k8s-v2-account

Application * ns-edu-zcp-01

Manifest Configuration

Manifest Source Text Artifacts

```
- apiVersion: apps/v1beta2
  kind: Deployment
  metadata:
    labels:
      app: recommendation-service
    name: recommendation-service-deployment-v3
    namespace: ns-zcp-edu-01
  spec:
    replicas: 1
    selector:
      matchLabels:
        app: recommendation-service
    template:
      metadata:
        labels:
          app: recommendation-service
      spec:
        containers:
          - env:
              - name: SPRING_PROFILES_ACTIVE
                value: k8s
            image: dtlabs/recommendation-service:latest
            imagePullPolicy: Always
            name: recommendation-service
            ports:
              - containerPort: 8080
```

- **Mainfest Configuration - Manifest Source: Text** 선택
- **Manifest:**

```
apiVersion: apps/v1beta2
kind: Deployment
metadata:
  labels:
    app: recommendation-service
    name: recommendation-service-deployment-v2
    namespace: '${parameters.namespace}'
spec:
  replicas: 1
  selector:
    matchLabels:
      app: recommendation-service
  template:
    metadata:
      labels:
        app: recommendation-service
        version: v2
    spec:
      containers:
        - env:
            - name: SPRING_PROFILES_ACTIVE
              value: k8s
            image: dtlabs/recommendation-service:latest
            imagePullPolicy: Always
            name: recommendation-service
            ports:
              - containerPort: 8080
      resources:
        limits:
          cpu: '0.5'
          memory: 1Gi
        requests:
          cpu: '0.25'
          memory: 256Mi
```

recommendation-service-deployment-v2.yaml

Docker hub triggering

5 Kubernetes에 Deployment 배포

- ✓ 생성된 파이프라인 확인

The screenshot shows a user interface for a continuous integration and deployment system. At the top, there's a navigation bar with tabs for 'Applications', 'Search', 'What's New', and 'Help'. Below the navigation, there are three main categories: 'PIPELINES' (with a red 'V' icon), 'INFRASTRUCTURE', and 'TASKS'. The 'TASKS' tab is selected, indicated by a checked checkbox icon. In the center, there are several filter and search options: '+', '-' buttons for grouping, a 'Group by Pipeline' dropdown set to 'Pipeline', a 'Show' dropdown set to '2', and checkboxes for 'executions per pipeline' and 'stage durations'. To the right of these are 'Create', 'Configure', and 'Start Manual Execution' buttons. Below this toolbar, a specific pipeline step is highlighted with a blue background and white text: 'Deploy recommendation V2'. To its right, it says 'Trigger: disabled' and has 'Configure' and 'Start Manual Execution' buttons. The overall interface is clean and modern, typical of DevOps tooling.

6 Docker Image 푸시

Docker Hub의 dtlabs/recommendation-service:2.0 Docker Image 푸시한다.

```
$ docker push dtlabs/recommendation-service:2.0
```

Docker hub triggering

Pipeline 실행 결과 확인

- ✓ Pipeline이 자동 시작하여 실행(RUNNING)

The screenshot shows a pipeline execution interface with the following details:

- Group by:** Pipeline
- Show:** 2 executions per pipeline
- Stage Durations:** Enabled
- Create**, **Configure**, and **Start Manual Execution** buttons

A stage named "Deploy recommendation v2" is listed, showing the following information:

- WEBHOOK:** a few seconds ago
- Status:** RUNNING
- Duration:** 00:08
- Details:** A button with a red arrow pointing to it, indicating the number of tasks running.

Below the stage details, there is a section titled "STAGE DETAILS: DEPLOY RECOMMENDATION" with the following table:

| Step | Started | Duration | Status |
|----------------------------|-------------------------|----------|---------|
| Deploy recommendation v3.0 | 2018-08-23 09:49:33 PDT | 00:08 | RUNNING |

Next to the table is a "DEPLOY RECOMMENDATION" card with tabs for Deploy Status, Task Status, and Artifact Status. The Deploy Status tab is active, showing a progress bar.

At the bottom right of the card, there are links for Source and Permalink.

Docker hub triggering

Pipeline 실행 결과 확인

Details를 클릭하여 현재 실행 단계에 대한 상세 정보를 확인할 수 있다.

Deploy recommendation 1 Trigger: enabled [Configure](#) [Start Manual Execution](#)

WEBHOOK a few seconds ago [Details](#) Status: **RUNNING** Duration: 00:46

Deploy recommendation v3.0

STAGE DETAILS: DEPLOY RECOMMENDATION Duration: 00:46

| Step | Started | Duration | Status |
|----------------------------|-------------------------|----------|----------------|
| Deploy recommendation v3.0 | 2018-08-23 09:49:33 PDT | 00:46 | RUNNING |

▶ DEPLOY RECOMMENDATION 현재 수행하고 있는 Task 상태 실시간 모니터링

| Deploy Status | Task Status | Artifact Status |
|---------------|--------------------|-----------------|
| | | |

| Task | Duration |
|--------------------------------|----------|
| Deploy Manifest | 00:00 |
| Monitor Deploy | 00:05 |
| Promote Outputs | 00:00 |
| Force Cache Refresh | 00:38 |
| Wait For Manifest To Stabilize | - |
| Cleanup Artifacts | - |
| Bind Produced Artifacts | - |

[Source](#) | [Permalink](#)

Docker hub triggering

Pipeline 실행 결과 확인

- ✓ Pipeline을 성공적으로 실행 완료되어 SUCCEEDED 표시된 결과 확인

The screenshot shows the CircleCI web interface with the following details:

- Dashboard:** Shows a pipeline named "ns-edu-zcp-01" with a "Deploy recommendation" step.
- Deploy recommendation Step:**
 - WEBHOOK:** Triggered 2 minutes ago.
 - Status:** SUCCEEDED (highlighted with a red arrow).
 - Duration:** 01:18
- STAGE DETAILS: DEPLOY RECOMMENDATION V3.0:**
 - Duration: 01:18
 - Step: Deploy recommendation v3.0 (Started: 2018-08-23 09:49:33 PDT, Duration: 01:18, Status: SUCCEEDED)
- DEPLOY RECOMMENDATION V3.0:**
 - Deployment:** recommendation-service-deployment-v3 (available, stable)
 - Tasks:**
 - 1 × ScalingReplicaSet (6 minutes ago: Scaled up replica set recommendation-service-deployment-v3-5ccf5976f5 to 1)
 - 1 × ScalingReplicaSet (4 minutes ago: Scaled down replica set recommendation-service-deployment-v3-5ccf5976f5 to 0)
 - 1 × ScalingReplicaSet (2 minutes ago: Scaled up replica set recommendation-service-deployment-v3-5ccf5976f5 to 1)
- Artifact Status:**
 - Produced Artifacts:** recommendation-service...

A red arrow points from the "Produced Artifacts" section to a pink box at the bottom right containing the text "Pipeline 실행하여 생성된 Kubernetes 오브젝트".

Docker hub triggering

Pipeline 실행 결과 확인

- ✓ Infrastructure에서 Workloads 확인

The screenshot shows the ns-edu-zcp-01 infrastructure dashboard with the following details:

- CLUSTERS:** MY-K8S-V2-ACCOUNT
- LOAD BALANCERS:** deployment recommendation-service-deployment- (1 instance, 100% healthy)
- FIREWALLS:** NS-ZCP-EDU-01
- SEARCH:** V001: dtlabs/recommendation-service:3.0
- INFORMATION:**
 - Created: 2018-08-23 09:49:34 PDT
 - Account: MY-K8S-V2-ACCOUNT
 - Namespace: ns-zcp-edu-01
 - Kind: deployment
 - Managing: replicaSet recommendation-service-deployment-v3-5ccf5976f5
- STATUS:**
 - Available: 2018-08-23T16:49:40Z
 - Progressing: 2018-08-23T16:49:34Z
- EVENTS:** 1 × ScalingReplicaSet (7 minutes ago)

Docker hub triggering

7 Docker Image Update

7 nginx Image로 deployment 업데이트

```
kubectl set image deployment/recommendation-service-deployment-v2 recommendation-service=nginx -n ${namespace name}

$ kubectl set image deployment/recommendation-service-deployment-v2 recommendation-service=nginx -n
${namespace name}
$ deployment.apps "recommendation-service-deployment-v2" image updated
```

✓ 업데이트 결과 확인

The screenshot shows the Spinnaker interface for managing Kubernetes resources. On the left, there's a list of clusters: CLUSTERS, LOAD BALANCERS, and FIREWALLS. The CLUSTERS section is active, showing 'MY-K8S-V2-ACCOUNT'. Below it, a table lists a single pod named 'recommendation-service-deployment-v3-86c79fc4-xnngc' from the 'NS-ZCP-EDU-01' namespace. The pod status is '1 ▲ / 1 ▼ : 50%' and it has a green icon. To the right, a detailed view of the deployment is shown in a modal window.

Deployment Details:

- Name:** recommendation-service-deployment
- Server Group Manager Actions:** (button)
- LABELS:** app: recommendation-service
- ANNOTATIONS:**
 - artifact.spinnaker.io/location: ns-zcp-edu-01
 - artifact.spinnaker.io/name: recommendation-service-deployment-v3
 - artifact.spinnaker.io/type: kubernetes/deployment
 - deployment.kubernetes.io/revision: 2
 - moniker.spinnaker.io/application: ns-edu-zcp-01
 - moniker.spinnaker.io/cluster: deployment recommendation-service-deployment-v3
- ARTIFACTS:** docker/image nginx

Docker hub triggering

8 Rollout

최초 버전으로 rollout한다

8-1 Deployment 상세 뷰의 **Server Group Manager Actions** 클릭

The screenshot shows the CloudBees DevCenter interface. On the left, there's a list of clusters: CLUSTERS, LOAD BALANCERS, and FIREWALLS. Under CLUSTERS, it shows 'MY-K8S-V2-ACCOUNT' and 'NS-ZCP-EDU-01'. In NS-ZCP-EDU-01, there are two instances: 'V003: dtlabs/recommendation-service:3.0' and 'V002: nginx'. A mouse cursor is hovering over the 'V003' instance. On the right, a detailed view of the 'recommendation-service-deployment' is shown. A dropdown menu titled 'Server Group Manager Actions' is open, with 'Undo Rollout' highlighted by a red arrow. The modal also lists 'Scale', 'Edit', and 'Delete' options.

8-2 Revision: Revision 1 선택, **Submit** 버튼 클릭

The screenshot shows a confirmation dialog titled 'Undo rollout of Deployment recommendation-service-deployment-v3 in ns-zcp-edu-01'. It has two fields: 'Revision' (set to 'Revision 1 (replicaSet recommendation-service-deployment-v3-5ccf6976f5)') and 'Reason' (a text input field containing '(Optional) anything that might be helpful to explain the reason for this change; HTML is okay'). At the bottom, there are 'Cancel' and 'Submit' buttons, with 'Submit' highlighted by a red arrow.

Docker hub triggering

8 Rollout

Undo rollout of deployment recommendation-service-deployment-v3 in ns-zcp-edu-01

- ✓ Undo Rollout Manifest (00:00)
- ✓ Monitor Undo Rollout (00:05)
- ✓ Force Cache Refresh (00:01)
- ✓ Wait For Stable (00:57)

Operation succeeded!

You can [monitor this task from the Tasks view.](#)

Rollout 실행 완료함

nginx가 적용된 pod가 없어지고 예전 Image가 적용된 pod가 생성됨을 확인

The screenshot shows the CloudBees CI/CD interface for the 'ns-edu-zcp-01' project. The 'CLUSTERS' tab is active. On the left, there's a sidebar with filters for 'SEARCH', 'ACCOUNT' (my-k8s-v2-account), 'REGION' (ns-zcp-edu-01), 'STACK' ((none)), 'DETAIL' ((none)), and 'STATUS'. The main area displays the 'recommendation-service-deployment' cluster under 'NS-ZCP-EDU-01'. It lists two pods: 'V003: dtlabs/recommendation-service:3.0' (green, 100% healthy) and 'V002: nginx' (red, 0% healthy). A modal window on the right provides detailed information about the deployment, including 'Server Group Manager Actions' for managing the service and replica set, and a 'STATUS' section indicating it is available.

Hands-on: Deploy Pipeline 생성하기



Docker Hub Triggering를 통한 Deploy

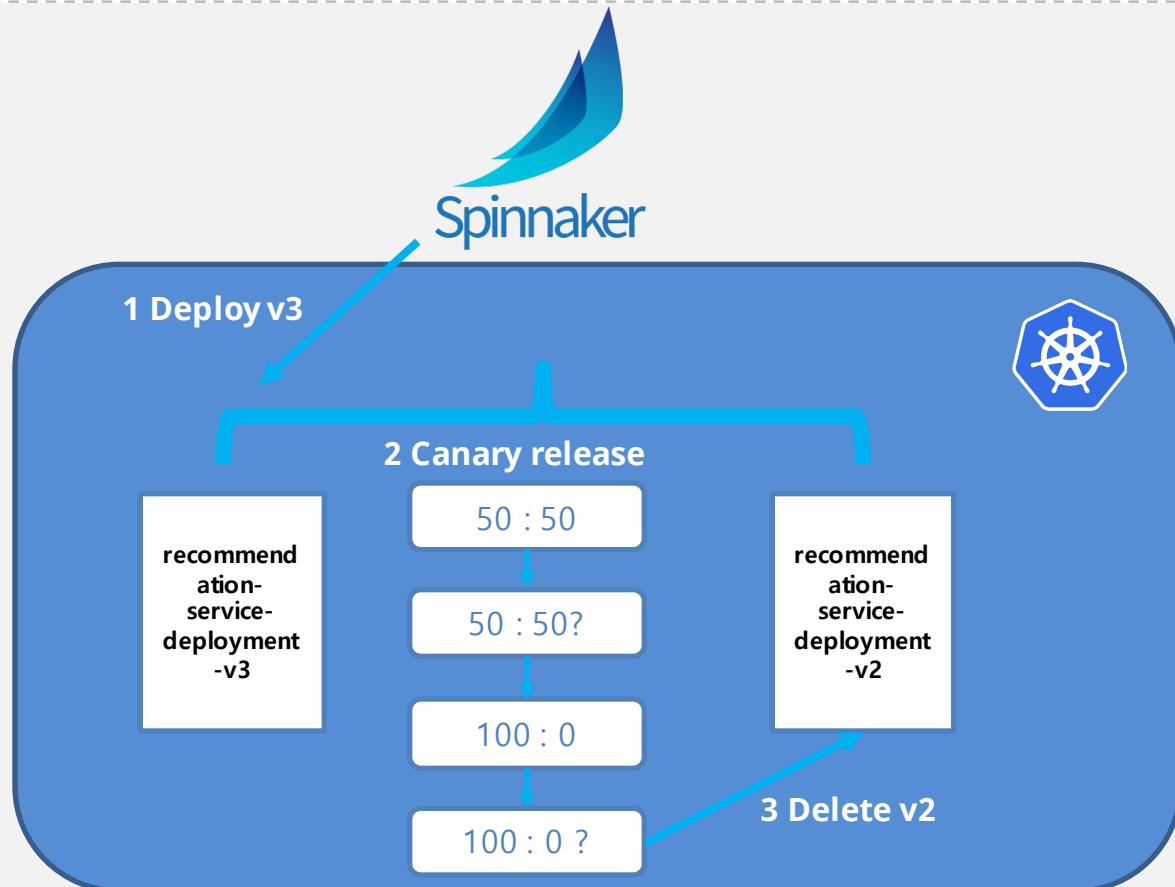


Canary release using istio

Canary release using istio

istio를 활용한 Canary release로 무중단 배포

시나리오

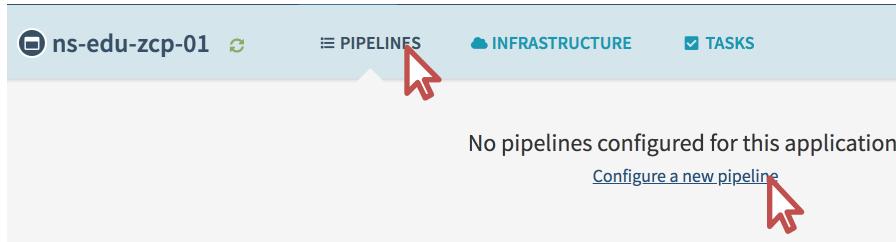


새 버전 애플리케이션을 canary release하고 기존 버전 삭제 (Parameter 사용)

Canary release using istio

1.2 Pipeline 생성, Parameters 적용

1-1 상단의 **PIPELINES** 클릭 – 하단의 **Configure a new pipeline** 클릭



1-2 Type: Pipeline 선택

Pipeline Name: 'Canary release' 입력
Create 버튼 클릭

2 Parameters 적용

parameter를 설정한다.

Configuration stages의 Parameters에서 3개를 다음과 같이 설정

- **Name:** namespace
- **Name:** app_old_version
- **Name:** app_new_version

| Parameters | | |
|----------------------------------|----------------------|---------------------------------------|
| <input type="button" value="↑"/> | Name | namespace |
| | Required | <input checked="" type="checkbox"/> |
| | Description | namespace 전체 |
| | Default Value | |
| <input type="checkbox"/> | Name | app_old_version |
| | Required | <input checked="" type="checkbox"/> |
| | Description | 기존 버전. ex.v2 |
| | Default Value | v2 |
| <input type="checkbox"/> | Name | app_new_version |
| | Required | <input checked="" type="checkbox"/> |
| | Description | label 값, deployment 이름 뒤에 붙는 버전 ex.v3 |
| | Default Value | v3 |

Canary release using istio

3 Kubernetes Deploy stage 생성하기

새로운 버전의 애플리케이션을 배포하는 stage를 생성한다.

3-1 상단의 **PIPELINES** 클릭 – 하단의 Configure a new pipeline

Stage Name: Deploy recommendation-new

3-2 하단의 **Deploy (Manifest) Configuration** 설정

recommendation-service-deployment.yaml 복사/붙여넣기

The screenshot shows the 'Deploy (Manifest) Configuration' screen. On the left, there's a sidebar with tabs: 'DEPLOY (MANIFEST) CONFIGURATION' (selected), 'EXECUTION OPTIONS', 'NOTIFICATIONS', 'COMMENTS', and 'PRODUCES ARTIFACTS'. The main area has two sections: 'Basic Settings' and 'Manifest Configuration'. In 'Basic Settings', 'Account' is set to 'my-k8s-v2-account' and 'Application' is set to 'ns-edu-zcp-01'. In 'Manifest Configuration', 'Manifest Source' is set to 'Text', indicated by a red arrow pointing to the 'Text' radio button. Below it, there's a large text area containing the YAML manifest. A large red arrow points from this text area to the right-hand side of the slide, where the full YAML code is displayed.

```

apiVersion: apps/v1beta2
kind: Deployment
metadata:
  labels:
    app: recommendation-service
  name: recommendation-service-deployment-${parameters.app_new_version}
  namespace: ${parameters.namespace}
spec:
  replicas: 1
  selector:
    matchLabels:
      app: recommendation-service
  template:
    metadata:
      labels:
        app: recommendation-service
        version: '${parameters.app_new_version}'
    spec:
      containers:
        - env:
            - name: SPRING_PROFILES_ACTIVE
              value: k8s
            image: dtlabs/recommendation-service:latest
            imagePullPolicy: Always
            name: recommendation-service
            ports:
              - containerPort: 8080
            resources:
              limits:
                cpu: '0.5'
                memory: 1Gi
              requests:
                cpu: '0.25'
                memory: 256Mi

```

```

apiVersion: apps/v1beta2
kind: Deployment
metadata:
  labels:
    app: recommendation-service
  name: recommendation-service-deployment-${parameters.app_new_version}
  namespace: ${parameters.namespace}
spec:
  replicas: 1
  selector:
    matchLabels:
      app: recommendation-service
  template:
    metadata:
      labels:
        app: recommendation-service
        version: '${parameters.app_new_version}'
    spec:
      containers:
        - env:
            - name: SPRING_PROFILES_ACTIVE
              value: k8s
            image: dtlabs/recommendation-service:latest
            imagePullPolicy: Always
            name: recommendation-service
            ports:
              - containerPort: 8080
            resources:
              limits:
                cpu: '0.5'
                memory: 1Gi
              requests:
                cpu: '0.25'
                memory: 256Mi

```

Canary release using istio

4.5 istio rounting 5:5 적용

5:5 비율로 라우팅하기 위해 istio 오브젝트를 배포한다.

4-1 Add stage 클릭

- **Stage Type: Deploy (Manifest)** 선택
Depends On: Deploy recommendation - new 선택
Stage Name: Deploy 5:5

4-2 배포할 Deployment의 매니페스트를 입력한다.

하단의 **Deploy (Manifest) Configuration** 설정

istio.yaml 복사/붙여넣기



```

apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: recommendation-service
  namespace: '${parameters.namespace}'
spec:
  host: recommendation-service
  subsets:
    - labels:
        version: '${parameters.app_old_version}'
        name: '${parameters.app_old_version}'
    - labels:
        version: '${parameters.app_new_version}'
        name: '${parameters.app_new_version}'
---
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: recommendation-service
  namespace: '${parameters.namespace}'
spec:
  hosts:
    - recommendation-service
  http:
    - route:
        - destination:
            host: recommendation-service
            subset: '${parameters.app_old_version}'
            weight: 50
        - destination:
            host: recommendation-service
            subset: '${parameters.app_new_version}'
            weight: 50

```

5 Manual Judgment

사용자가 수동으로 5:5로 라우팅이 되는지 확인하는 stage를 추가

Add stage 클릭 - **Stage Type: ManualJudgement**선택

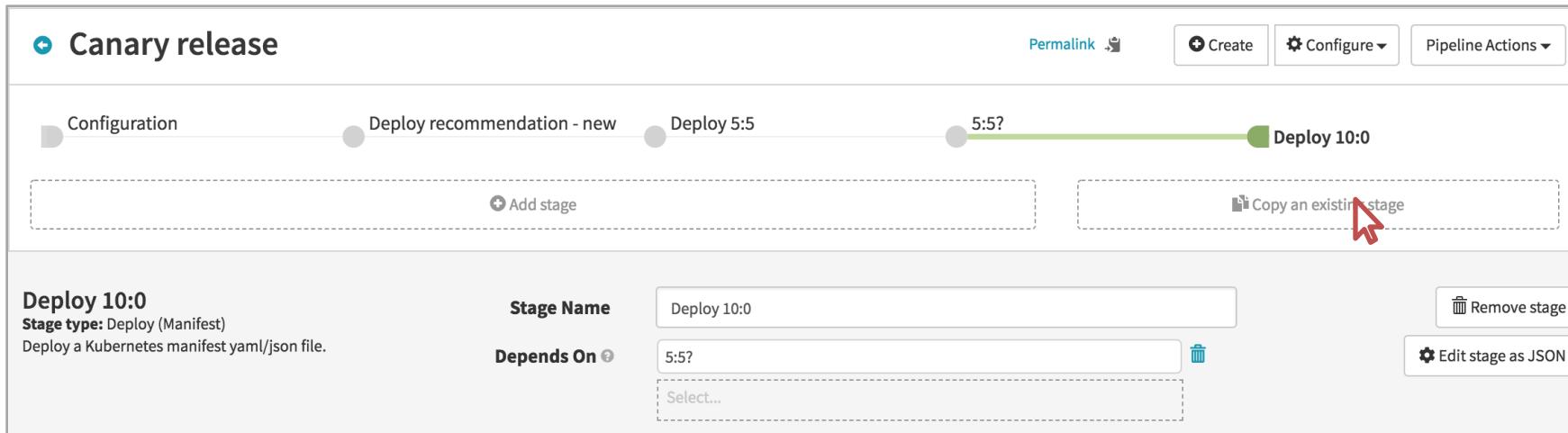
- Stage Name:** 5:5?
- Depends On:** Deploy 5:5 선택

Canary release using istio

6 istio rounting 10:0 적용

10:0 비율로 라우팅하기 위해 istio 오브젝트 배포한다.
이전에 만든 'Deploy 5:5' stages를 복사하여 사용한다.

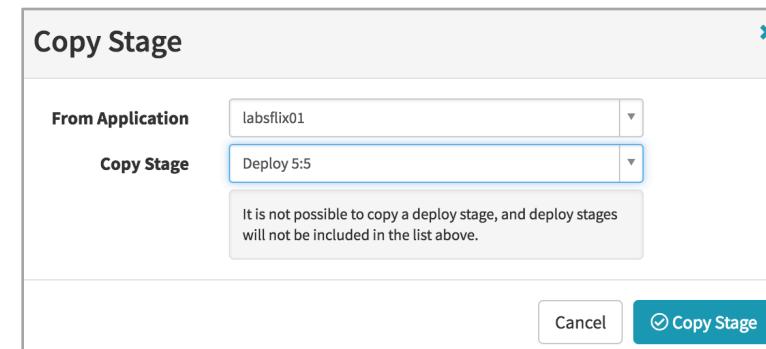
6-1 Copy an existing stage 클릭



The screenshot shows the 'Canary release' pipeline configuration. At the top, there is a horizontal timeline with stages: Configuration, Deploy recommendation - new, Deploy 5:5, 5:5?, and Deploy 10:0. Below the timeline, there are two dashed boxes. The left box contains a '+ Add stage' button. The right box contains a 'Copy an existing stage' button, which is highlighted with a red arrow pointing to it. In the main configuration area, there is a 'Deploy 10:0' stage defined with a Stage Name of 'Deploy 10:0' and a 'Depends On' field containing '5:5?'. To the right of this stage are buttons for 'Remove stage' and 'Edit stage as JSON'.

6-2 Copy Stage 팝업창에서 다음과 같이 선택

- **From Application:** 현재 application 선택
- **Copy Stage:** Deploy 5:5



Canary release using istio

6.7 istio rounting 10:0 적용

6-3 배포할 Deployment의 매니페스트를 수정한다.

하단의 **Deploy (Manifest) Configuration** 설정

subset: \${parameters.app_old_version}의 weight → 0 수정
 subset: \${parameters.app_new_version}의 weight → 100 수정

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: recommendation-service
  namespace: '${parameters.namespace}'
spec:
  hosts:
    - recommendation-service
  http:
    - route:
        - destination:
            host: recommendation-service
            subset: '${parameters.app_old_version}'
            weight: 0
        - destination:
            host: recommendation-service
            subset: '${parameters.app_new_version}'
            weight: 100
```

7 Manual Judgment

사용자가 수동으로 10:0로 라우팅이 되는지 확인하는 stage를 추가

Add stage 클릭 – **Stage Type: ManualJudgement** 선택

Stage Name: 10:0?

Depends On: Deploy 10:0 선택

Canary release using istio

8 Deployment old version 삭제

예전 버전의 애플리케이션을 삭제한다.

8-1 Add stage 클릭

- **Stage Type: Deletey (Manifest)** 선택
- **Depends On:** 10:0?선택

8-2 하단의 Manifest를 다음과 같이 설정

- **Namespace:** \${parameters.namespace}
- **Match On:** Name 선택
- **Kind:** deployment
- **Name:** recommendation-service-deployment-\${parameters.app_old_version}

Delete (Manifest) Configuration

Warning! This stage is under active development and is subject to change.

Manifest

| | |
|------------------|--|
| Account | my-k8s-v2-account |
| Namespace | \${parameters.namespace} |
| Match On | <input type="radio"/> Labels <input checked="" type="radio"/> Name |
| Kind | deployment |
| Name | recommendation-service-deployment-\${parameters.app_old_version} |

Canary release using istio

Pipeline 실행 결과 확인

1 Pipeline 화면에서 Canary release pipeline 수동 실행

The screenshot shows the 'INFRASTRUCTURE' tab selected in the top navigation bar. Below it, there are buttons for 'Group by Pipeline' and 'Show 2 executions per pipeline'. On the right, there are 'Create', 'Configure', and 'Start Manual Execution' buttons. The 'Canary release' pipeline is listed, and its details show 'Trigger: disabled'. A red arrow points to the 'Start Manual Execution' button.

2 Confirm Execution 설정

- **namespace**: 본인의 namespace명 전체 입력

3 Run 버튼 클릭

This will start a new run of **Canary release**.

This pipeline is parameterized. Please enter values for the parameters below.

| | |
|-------------------|----------------------|
| namespace * | <input type="text"/> |
| app_old_version * | v2 |
| app_new_version * | v3 |

* Required

Notifications Notify me when the pipeline completes

Cancel **Run**

감사합니다

