

k0s등 오픈소스로 kubernetes 개발환경 손쉽게 구성하기

OpenInfra Days Korea 2025

황인환



Table of contents

01

K0S란?

Why k0s?

Compare other k8s distribution

02

FluxCD 알아보기

Gitless GitOps, OCI

03

Operator Pattern

olm, operator framework

Sample operator

04

Hands-on

k0s 설치부터 gitops 구성까지



About Me



황인환

SK C&C (2012 ~ 2018)

삼성전자(2018 ~ 2020)

신한카드(2020 ~ 2025)

Microsoft(2025 ~)



Family

10,6세

온가족이 두산팬❤️



최고의 순간

Won 3rd prize at the
CNCF's first
CloudNativeHacks
Hackathon



kubestronaut

passing all five of
CNCF's
Kubernetes-related
certifications
(2025.07 ~)

What is this topic about?



K0S

K0s가 무엇인가?
K0s가 왜 특별한가?
K0s를 어떻게
사용하는가?



operator

Operator
framework란
Operator 사용법



FluxCD

Flux란?
Fluxcd가 왜
특별한가?
Github와 fluxcd를
활용한 gitops 패턴

~~Kubernetes deep dive, GitOps deep dive~~

01

K0S

Why k0s?
Compare other k8s distribution

k0s

The screenshot shows the Cloud Native Landscape page for k0s. The page is titled 'k0s' and is categorized under 'CNCF' and 'SANDBOX'. It features a maturity timeline with stages: SANDBOX (2025-01-19), INCUBATING, and GRADUATED. Below the timeline, there are statistics for k0s, including 5.3k stars, 133 contributors, and the latest commit and release dates.

Cloud Native Landscape

EXPLORE GUIDE STATS

Type / to search items

k0s CNCF SANDBOX 2025

Cloud Native Computing Foundation (CNCF)

Orchestration & Management Scheduling & Orchestration

k0s - The Zero Friction Kubernetes

ADDITIONAL CATEGORIES

Platform / Certified Kubernetes - Distribution

Maturity

2025-01-19 SANDBOX INCUBATING GRADUATED

Repositories

k0sproject/k0s (primary)

<https://github.com/k0sproject/k0s> PRIMARY Other good first issues 9 open

5.3k Stars

133 Contributors

Jun '20 First commit

Aug '25 Latest commit

Jul '25 Latest release

2025.01.19 – CNCF sandbox

The Zero Friction Kubernetes

- Any cloud
- Bare metal
- Edge and IoT

With k0s new clusters can be bootstrapped in minutes and developer friction is reduced to zero.

5.1k github stars(8월 현재)

k0s



Key Features

- **Certified and 100% upstream Kubernetes**
- Multiple installation methods: single-node, multi-node, airgap and Docker
- **Automatic lifecycle management with k0sctl: upgrade, backup and restore**
- Modest system requirements (1 vCPU, 1 GB RAM)
- **Available as a single binary with no external runtime dependencies besides the kernel**
- **Supports custom Container Network Interface (CNI) plugins (Kube-Router is the default, Calico is offered as a preconfigured alternative)**
- **Supports all Kubernetes storage options with Container Storage Interface (CSI)**

Why k0s?

Motivation#

A gap between the host OS and Kubernetes that runs on top of it

문제: OS와 K8S가 독립적으로 업그레이드되며 취약점·성능 문제 책임이 불명확.

해결: k0s는 커널 외 의존성이 없는 단일 바이너리로 제공 → 모든 취약점·성능 이슈를 k0s에서 해결.

K8S with partial FIPS security compliance

문제: 기존 K8S는 일부만 FIPS 준수 → 중요 애플리케이션 보안 리스크.

해결: k0s는 코어부터 포함된 모든 컴포넌트를 100% FIPS 준수 빌드 가능.

Kubernetes with cumbersome lifecycle management

문제: 높은 시스템 요구사항, OS/인프라 종속성, 다양한 유즈케이스 대응 어려움.

해결: k0s는 경량 설계 + 자동화 관리 도구 제공, 어떤 OS·인프라에서도 동작하며 엣지·IoT·클라우드 등 다양한 환경 확장 가능.

Compare other k8s distribution

Provides a robust and versatile “base”

k0s는 불필요한 애드온을 최소화하고, 견고하고 유연한 Kubernetes 기반 제공.

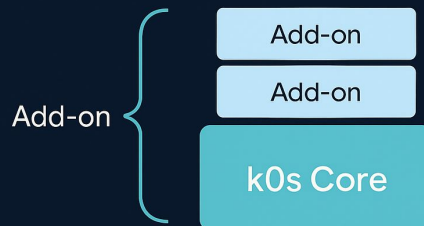
Minimizes bundled add-ons

외부 애드온을 많이 포함하면 업스트림 릴리스 추적이 어렵고 구버전 제공은 의미 없음.

Avoids opinionated choices

Ingress, Service Mesh, Storage 등은 매우 의견이 강함 → 사용자가 선택하도록.

Lightweight Base Architecture



- Provides a robust Kubernetes “base”
- Minimizes bundled add-ons
- Avoids opinionated choices

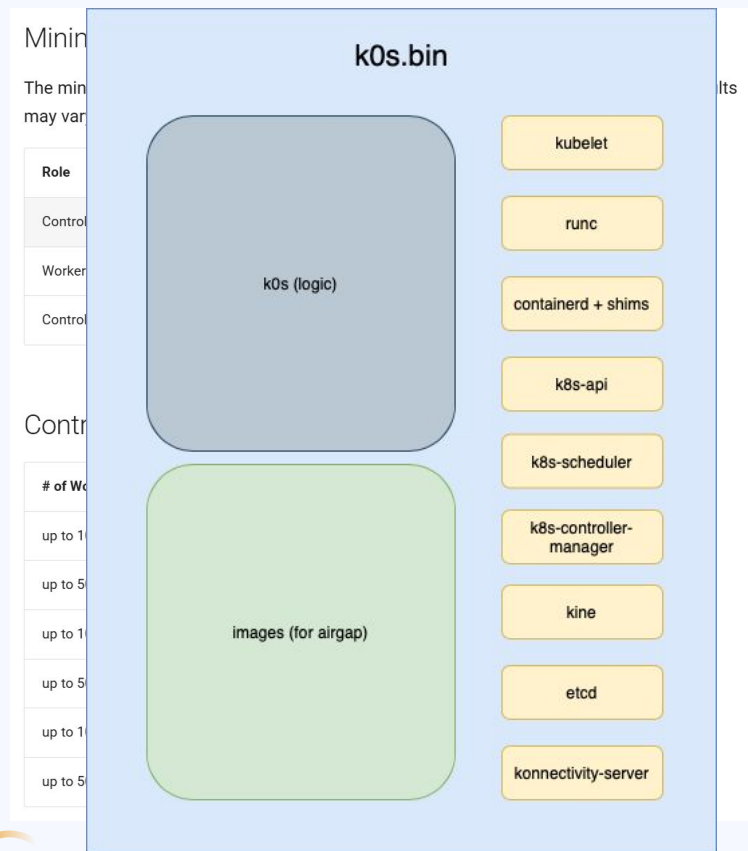
Compare other k8s distribution

vs k8s

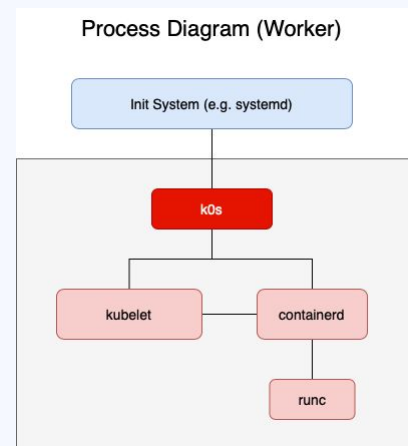
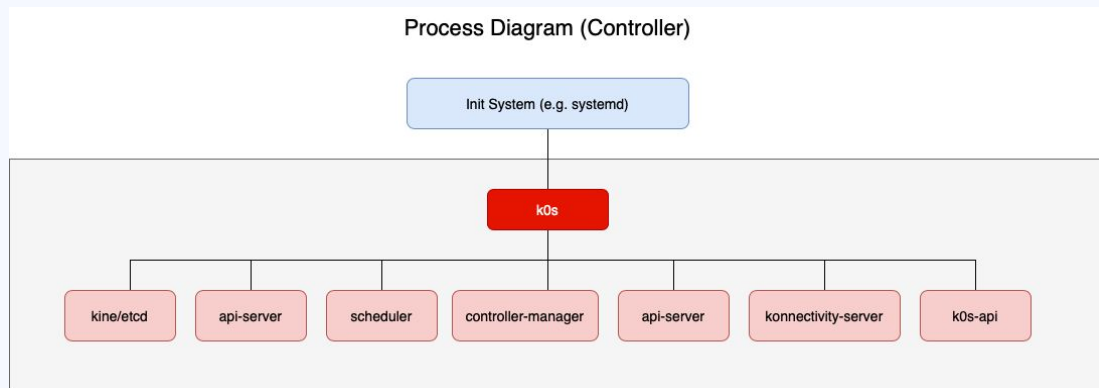
- 최소한의 의존성을 지닌 lightweight Kubernetes
- 단일 바이너리(k0s)를 통한 손쉬운 설치(vs kubeadm)

vs other lightweight Kubernetes distribution(like k0s, microk8s)

- cli(k0scli)와 yml을 통한 배포 지원
- Devops와 통합하기 용이



Compare other k8s distribution

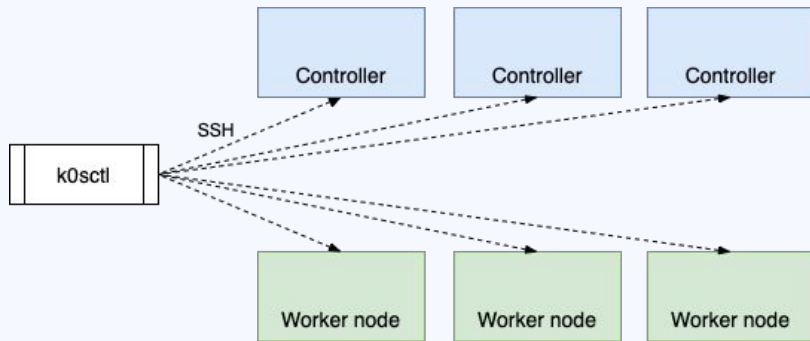


As a single binary, k0s acts as the process supervisor for all other control plane components. As such, there is no container engine or kubelet running on controllers by default, which thus means that a cluster user cannot schedule workloads onto controller nodes.

As with the control plane, with k0s you can create and manage the core worker components as naked processes on the worker node.

k0sctl

```
apiVersion: k0sctl.k0sproject.io/v1beta1
kind: Cluster
metadata:
  name: k0s-cluster
spec:
  hosts:
    - role: controller
      ssh:
        address: 10.0.0.1 # replace with the controller's IP address
        user: root
        keyPath: ~/.ssh/id_rsa
    - role: worker
      ssh:
        address: 10.0.0.2 # replace with the worker's IP address
        user: root
        keyPath: ~/.ssh/id_rsa
```



k0sctl apply --config k0sctl.yaml

command-line tool 로 kubernetes를 설치/업데이트/수정 등 작업을 진행할 수 있음
YAML 기반 설정 정보를 통하여 진행 -> git으로 통합 관리 가능성

k0sctl

```
apiVersion: k0sctl.k0sproject.io/v1beta1
kind: Cluster
metadata:
  name: k0s-cluster
spec:
  hosts:
    # Controller 노드 1대
    - ssh:
        address: 10.0.0.7
        user: azureuser
        keyPath: ~/.ssh/bastion_key
        role: controller
        hostname: controller-1
        installFlags:
          - "--enable-metrics-scraper"
    - ssh:
        address: 10.0.0.4
        user: azureuser
        keyPath: ~/.ssh/bastion_key
        role: worker
        hostname: worker-1
  k0s:
    version: v1.33.2+k0s.0
    # version: v1.33.3+k0s.0
    config:
      apiVersion: k0s.k0s.io/v1beta1
      kind: ClusterConfig
      metadata:
        name: k0s
      spec:
        # connectivity port 기본설정
        connectivity:
          adminPort: 8133
          agentPort: 8132
        network:
          provider: calico
          calico:
            envVars:
              CALICO_IP4POOL_CIDR: "10.244.0.0/16"
              CALICO_DISABLE_FILE_LOGGING: "true"
              FELIX_DEFAULTENDPOINTACTION: "ACCEPT"
              FELIX_LOGSEVERITYSCREEN: "info"
              FELIX_HEALTHENABLED: "true"
              FELIX_PROMETHEUSCONFIGENABLED: "true"
              FELIX_FEATUREDETECTOVERRIDE: "ChecksumOffloadBroken=true"
              FELIX_IPMASQUERADE: "false"
        extensions:
          helm:
            concurrencyLevel: 5
            repositories:
              - name: stable
                url: https://charts.helm.sh/stable
              - name: prometheus-community
                url: https://prometheus-community.github.io/helm-charts
              - name: grafana
                url: https://grafana.github.io/helm-charts
            charts:
              - name: kps
                chartName: prometheus-community/kube-prometheus-stack
                version: "76.2.0"
                namespace: monitoring
```

Sample k0sctl yamI file

- 노드 정보 추가/확장 가능
- 컴포넌트 선택(cni)
- Helm chart build-in install 기능으로 csi 등 확장 기능 관리

More...

Airgapped Installation

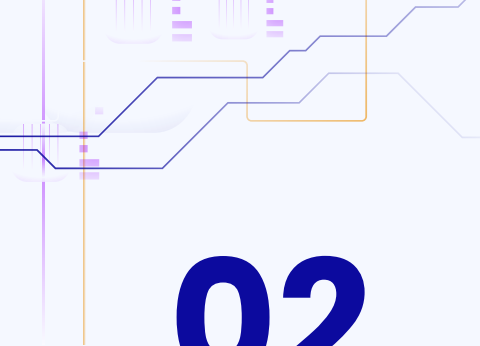
- install k0s in environments without Internet access

Autopilot

- Automatic updates
- safeguards in place to avoid breaking a cluster
- Status Reporting

Extensions

- MetalLB Load Balancer
- Ingress Controller(Nginx/Traefik)
- GitOps with Flux
- Ceph Storage with Rook
- OpenEBS storage



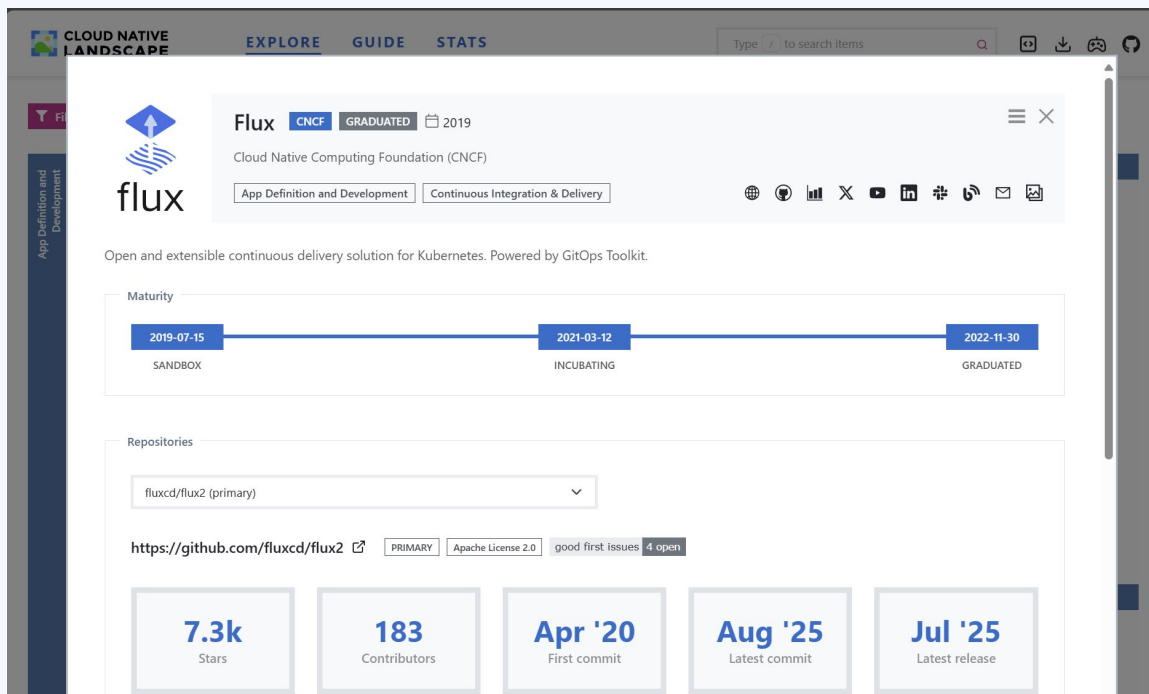
02

FluxCD 알아보기

Gitless GitOps, OCI



Flux



2019.07.15 – CNCF sandbox
2021.03.12 – CNCF incubating
2022.11.30 – CNCF graduated

Open and extensible continuous delivery solution for Kubernetes. Powered by GitOps Toolkit.

Flux is a set of continuous and progressive delivery solutions for Kubernetes that are open and extensible.

7.3k github stars(8월 현재)

Flux vs ArgoCD

FluxCD

- 경량, 모듈화, CLI 중심
- → DevOps 팀이 GitOps 표준을 단순하게 구현할 때 적합

ArgoCD

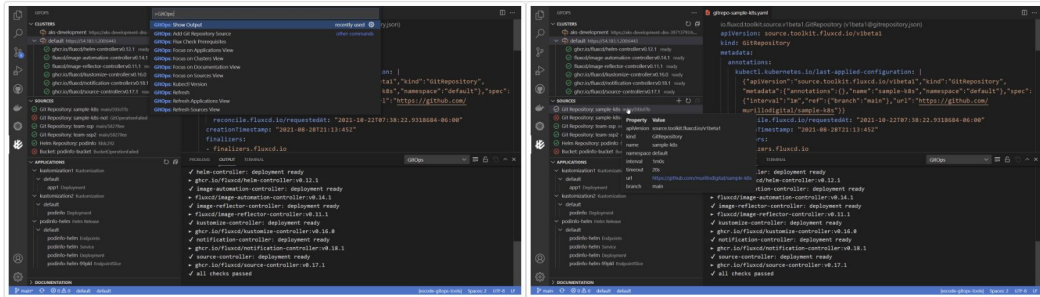
- UI/UX 강점, 시각화, 멀티 클러스터 관리
- → 운영팀, 시각화 요구가 큰 환경에 적합

Flux

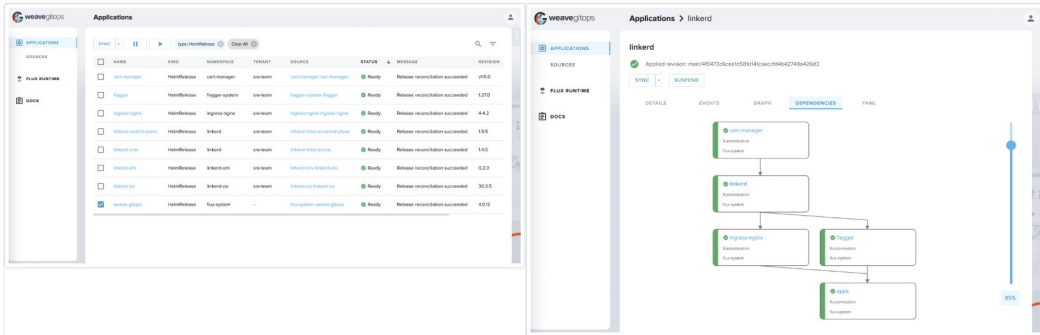
- 🔧 Flux provides GitOps for both apps and infrastructure
 - Flux and [Flagger](#) **deploy apps with canaries, feature flags, and A/B rollouts**. Flux can also manage any Kubernetes resource. Infrastructure and workload dependency management is built in.
- 📦 **Just push to Git and Flux does the rest**
 - Flux enables application deployment (CD) and (with the help of [Flagger](#)) progressive delivery (PD) through automatic reconciliation. Flux can even push back to Git for you with automated container image updates to Git (image scanning and patching).
- 🔗 Flux works with your existing tools
 - Flux works with your Git providers (GitHub, GitLab, Bitbucket, can even use s3-compatible buckets as a source), all major container registries, fully integrates [with OCI](#) and all CI workflow providers.
- 🔒 Flux is designed with security in mind
 - Pull vs. Push, least amount of privileges, adherence to Kubernetes security policies and tight integration with security tools and best-practices. Read more about [our security considerations](#).
- ⚙️ Flux works with any Kubernetes and all common Kubernetes tooling
 - Kustomize, Helm, RBAC, and policy-driven validation (OPA, Kyverno, admission controllers) so it simply falls into place.
- 👥 Flux does Multi-Tenancy (and “Multi-everything”)
 - Flux uses true Kubernetes RBAC via impersonation and supports multiple Git repositories. Multi-cluster infrastructure and apps work out of the box with Cluster API: Flux can use one Kubernetes cluster to manage apps in either the same or other clusters, spin up additional clusters themselves, and manage clusters including lifecycle and fleets.
- 📊 Dashboards love Flux
 - No matter if you use one of [the Flux UIs](#) or a hosted cloud offering from your cloud vendor, Flux has a thriving ecosystem of integrations and products built on top of it and all have great dashboards for you.
- 📞 Flux alerts and notifies
 - Flux provides health assessments, alerting to external systems, and external events handling. Just “git push”, and get notified on Slack and [other chat systems](#).
- 👍 Users trust Flux
 - Flux is a CNCF Graduated project and was categorised as “Adopt” on the [CNCF CICD Tech Radar](#)** (alongside Helm).
- 💖 Flux has a lovely community that is very easy to work with!
 - We welcome contributors of any kind. The components of Flux are on Kubernetes core controller-runtime, so anyone can contribute and its functionality can be extended very easily.

Flux UIs/ GUIs

VS Code GitOps Tools



Weave GitOps



Built-in UI는 제공되지 않으며,
weaveworks에서 제공하는 vs
code extension과 weave gitops
dashboard가 잘 알려져 있음
(opensource)

그 외.

- [gimlet-io/capacitor](#)
- [headlamp/flux-plugin](#)
- [freelensapp/freelens-fluxcd-e](#)
[xtension](#)
- [vmware-tanzu/kubeapps](#)

Flux – Gitless GitOps

기원

- 2022년 Flux 팀이 OCIRepository 소스 타입과 **Flux OCI Artifact** 미디어 타입 도입
- GitOps에서 Git 의존성 제거, OCI 레지스트리를 단일 신뢰 소스로 활용

특징

- Flux 컨트롤러는 Git과 완전 분리 → OCI 레지스트리만 참조
- 사용자는 여전히 Git 인터페이스 활용 (PR, 코드 리뷰 등)
- 구성 아티팩트, SBOM, 서명, 앱 이미지 모두 OCI 레지스트리에 저장

장점

- 유연성·확장성 향상: Git 서버는 더 이상 운영 필수 아님
- 단일 소스 오브 트루스: OCI 레지스트리 중심 관리
- 보안·서명·배포 통합 가능

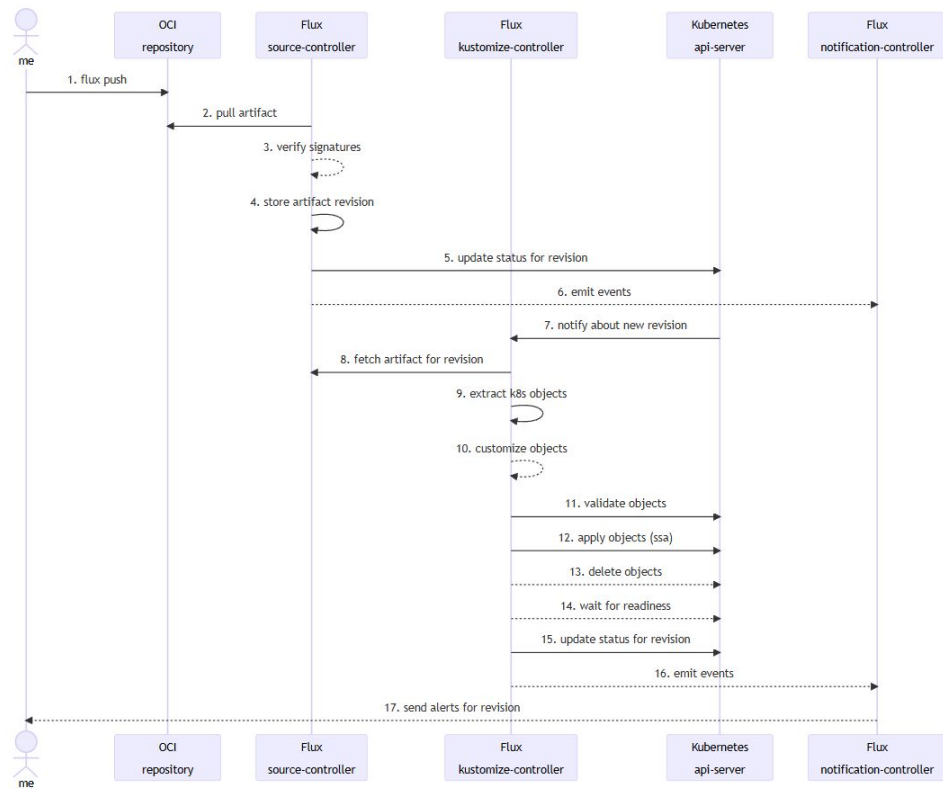
Flux – Gitless GitOps

```
apiVersion: source.toolkit.fluxcd.io/v1
kind: OCIRepository
metadata:
  name: podinfo
  namespace: default
spec:
  interval: 10m
  url: oci://ghcr.io/stefanprodan/charts/podinfo
  layerSelector:
    mediaType: "application/vnd.cncf.helm.chart.content.v1.tar+gzip"
    operation: copy
  ref:
    semver: ">=6.9.0"
---
apiVersion: helm.toolkit.fluxcd.io/v2
kind: HelmRelease
metadata:
  name: podinfo
  namespace: default
spec:
  interval: 10m
  releaseName: podinfo
  chartRef:
    kind: OCIRepository
    name: podinfo
  values:
    replicaCount: 2
```

이미지 배포 시 **helm** 등 배포가 필요한 **object**를 추가하고 이 정보를 메타데이터에 정의한다.

Flux는 정해진 스케줄마다 OCI Repository에서 이미지를 받아온다.

Flux - Gitless GitOps



이미지 배포 시 **helm** 등 배포가 필요한 **object**를 추가하고 이 정보를 메타데이터에 정의한다.

Flux는 정해진 스케줄마다 OCI Repository에서 이미지를 받아온다.

Flux는 들고 온 이미지를 풀어 **helm chart install** 등 지정된 스펙에 맞추어 **CD** 작업을 수행한다.

CD 수행 시 **git**에 대한 의존성이 사라지고 **oci repository**에 대해서만 의존한다.



03

Operator Pattern

olm, operator framework, Sample operator





"Kubernetes' operator pattern concept lets you extend the cluster's behaviour without modifying the code of Kubernetes itself by linking controllers to one or more custom resources. Operators are clients of the Kubernetes API that act as controllers for a Custom Resource."

Kubernetes Operator Pattern

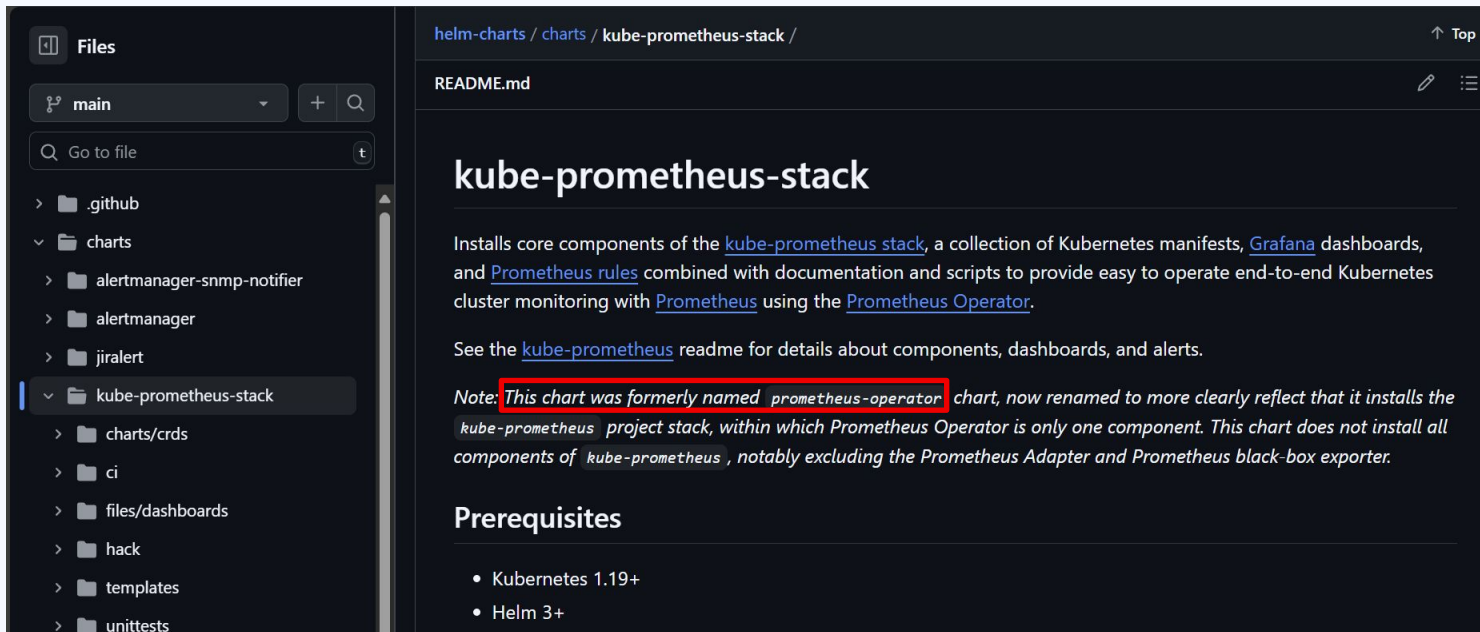


Operator vs Helm

관점	Helm Chart	Kubernetes Operator	한 줄 요약
운영 모델	매니페스트 패키징/배포 도구	지속적 Reconcile로 운영 자동화	설치 도구 vs 운영 컨트롤러
상태/자가치유	일회성 적용, 드리프트 자동 교정 없음	상태 감시로 드리프트 자동 교정/복구	안정성은 Operator 우위
Day-2 자동화	Hook로 제한적(업그레이드 스크립트 수준)	백업/복구/페일오버/마이그 등 로직 내장 가능	복잡 운영은 Operator
복잡도/비용	낮음(빠른 온보딩)	높음(개발/운영 투자 요구)	초기 투자 vs 장기 효율
권한/보안	실행 시점 권한 최소화 가능	컨트롤러에 지속 권한 필요	보안 표면은 Operator가 넓음

Operator

Operator는 단일 YAML 파일 또는 helm chart를 통한 설치가 일반적임.



Files

main

Go to file

- .github
- charts
 - alertmanager-snmp-notifier
 - alertmanager
 - jiralert
 - kube-prometheus-stack**
 - charts/crds
 - ci
 - files/dashboards
 - hack
 - templates
 - unittests

helm-charts / charts / kube-prometheus-stack /

↑ Top

README.md

kube-prometheus-stack

Installs core components of the [kube-prometheus stack](#), a collection of Kubernetes manifests, [Grafana](#) dashboards, and [Prometheus rules](#) combined with documentation and scripts to provide easy to operate end-to-end Kubernetes cluster monitoring with [Prometheus](#) using the [Prometheus Operator](#).

See the [kube-prometheus](#) readme for details about components, dashboards, and alerts.

*Note: This chart was formerly named **prometheus-operator** chart, now renamed to more clearly reflect that it installs the `kube-prometheus` project stack, within which Prometheus Operator is only one component. This chart does not install all components of `kube-prometheus`, notably excluding the Prometheus Adapter and Prometheus black-box exporter.*

Prerequisites

- Kubernetes 1.19+
- Helm 3+

Operator

Prometheus Operator Docs Kube-Prometheus Adopters

> GETTING-STARTED

> API REFERENCE

▼ PLATFORM GUIDE

- Getting Started
- Exposing Prometheus and Alertmanager
- Admission webhook
- Prometheus Agent
- Thanos
- RBAC
- RBAC for CRDs
- High Availability
- Storage
- Strategic Merge Patch
- CLI reference
- Troubleshooting

> DEVELOPER GUIDE

> PROPOSALS

> COMMUNITY

Apply all these manifests to create the necessary RBAC resources. Now you are all set to deploy a Prometheus instance. Here is an example of a basic Prometheus instance manifest.

```
apiVersion: monitoring.coreos.com/v1
kind: Prometheus
metadata:
  name: prometheus
spec:
  serviceAccountName: prometheus
```

To verify that the instance is up and running, run:

```
kubectl get -n default prometheus prometheus -w
```

For more information, see the Prometheus Operator RBAC guide.

Deploying Alertmanager

Let us take a simple example that creates 3 replicas of Alertmanager.

```
apiVersion: monitoring.coreos.com/v1
kind: Alertmanager
metadata:
  name: example
spec:
  replicas: 3
```

Operator를 설치해 놓으면 손쉬운 yaml 파일 하나만으로도 application (예: Prometheus, alertmanater 등) 배포가 가능해짐.

또한 좌측에서 배포한 CRD object(Prometheus)를 삭제하면 연관된 Application이 일괄 삭제 가능

Operator

Operator Framework를 활용하면 배포 가능한 operator를 만들어 활용할 수 있음.

 OPERATOR SDK

[Home](#) [Build](#) [Documentation](#) [Releases](#)

INSTALLING THE SDK CLI

Follow the steps in the [installation guide](#) to learn how to install the Operator SDK CLI tool. If you are using a release version of the SDK, make sure to follow the documentation for that version. You make use any of the following installation processes:

- [Install the Homebrew \(macOS\)](#)
- [Install from GitHub release](#)
- [Compile and install from master](#)

[Learn More](#) →

HOMEBREW, GITHUB, MASTER.

GO ANSIBLE HELM

READ THE USER GUIDES

Operators can be created with the SDK using Ansible, Helm, or Go. Follow the one of the quickstart guides to dive in.

- [Ansible Quickstart](#) →
- [Helm Quickstart](#) →
- [Go Quickstart](#) →

← ↻ <https://github.com/hellices/openapi-aggregator-operator>

README Apache-2.0 license

OpenAPI Aggregator Operator

go report [Alt](#) license [Apache 2.0](#) Go [v1.22.0](#)

Kubernetes operator that discovers and aggregates OpenAPI/Swagger specifications from services running in your cluster. It provides a unified Swagger UI interface to browse and test all your APIs in one place.

Quick Start

1. Installation

```
# Install the operator
kubectl apply -f https://raw.githubusercontent.com/hellices/openapi-aggregator-operator/main/install

# Verify the installation
kubectl get pods -n openapi-aggregator-system
```

2. Configure Services

Add these annotations to your services that expose OpenAPI/Swagger specs:

```
metadata:
  annotations:
    openapi.aggregator.io/swagger: "true" # Required
    openapi.aggregator.io/path: "/v2/api-docs" # Optional (default: /v2/api-docs)
    openapi.aggregator.io/port: "8080" # Optional (default: 8080)
    openapi.aggregator.io/allowed-methods: "get,post" # Optional: Filter allowed HTTP methods
```

04

시연

K0s, fluxcd, operator 등