K8s management with GitOps

Paweł Wichary

Agenda

- Prerequisites & disclaimer,
- Introduction to Kustomize,
- Lightweight configuration management,
- Introduction to FluxCD,
- Setup K8s with FluxCD and Kustomize,

Prerequisites & disclaimer

This presentation is not intended to present the best & most complex approach.
The main goal is to show developer friendly way for K8s management.

Prerequisites:

- Basic knowledge of Kubernetes (K8s)
- Basic knowledge of Git

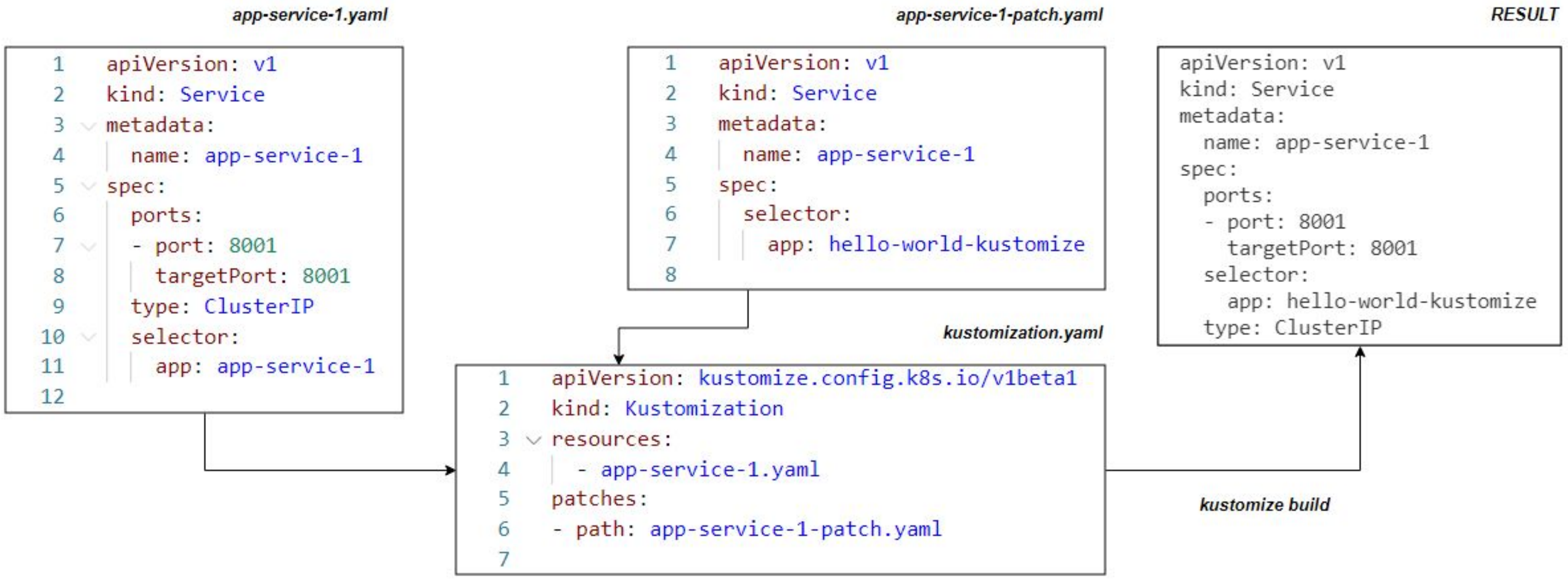
Introduction to Kustomize

Kustomize is a tool for declarative management of Kubernetes objects. It lets you customize raw, template-free YAML files for multiple purposes, leaving the original YAML untouched and usable as is.

Project page → https://kustomize.io/

GitHub repo → https://github.com/kubernetes-sigs/kustomize/

How it works?



app-service-1.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: app-service-1
spec:
  ports:
  - port: 8001
    targetPort: 8001
  type: ClusterIP
  selector:
    app: app-service-1

```

app-service-1-patch.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: app-service-1
spec:
  selector:
    app: hello-world-kustomize

```

kustomization.yaml

```yaml
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization
resources:
  - app-service-1.yaml
patches:
- path: app-service-1-patch.yaml

```

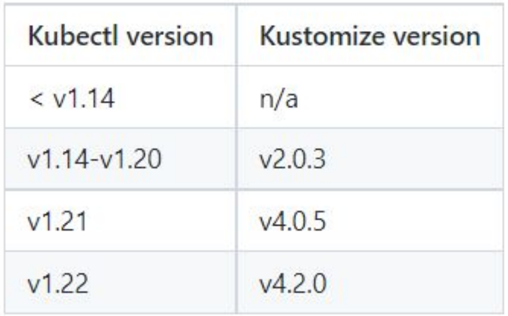
RESULT

```yaml
apiVersion: v1
kind: Service
metadata:
  name: app-service-1
spec:
  ports:
  - port: 8001
    targetPort: 8001
  selector:
    app: hello-world-kustomize
  type: ClusterIP
```

Install Kustomize

Install Kustomize in two ways:

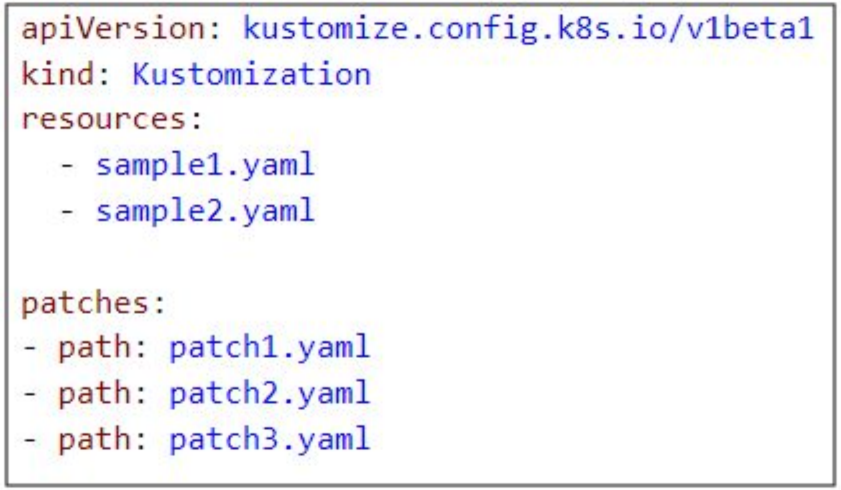
- as standalone app
  - https://kubectl.docs.kubernetes.io/installation/kustomize/
- with kubectl
  - https://github.com/kubernetes-sigs/kustomize#kubectl-integration



Kubectl version	Kustomize version
< v1.14	n/a
v1.14-v1.20	v2.0.3
v1.21	v4.0.5
v1.22	v4.2.0

Getting started

Kustomization is basically a file that allows to point out other files containing kubernetes objects. It is possible to merge and modify configuration using the Kustomize syntax.



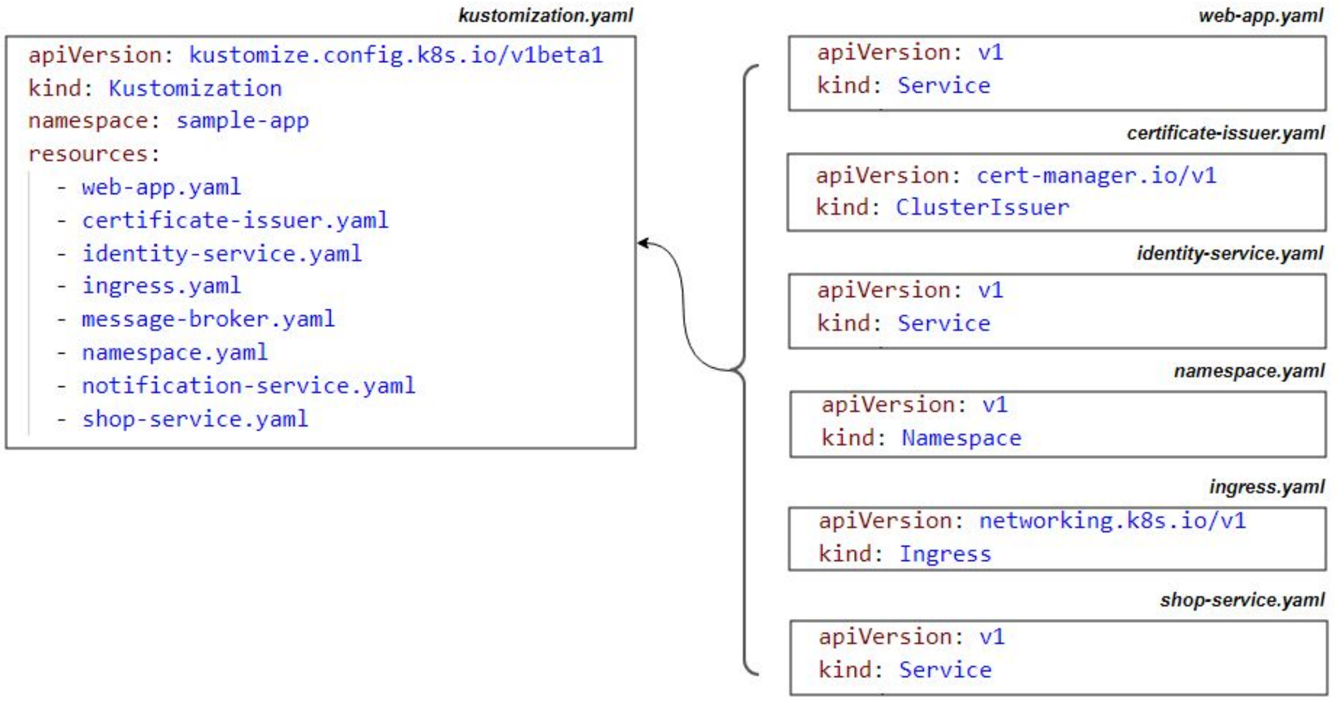
```yaml
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization
resources:
  - sample1.yaml
  - sample2.yaml

patches:
- path: patch1.yaml
- path: patch2.yaml
- path: patch3.yaml
```

Kustomization API Reference: https://kubectl.docs.kubernetes.io/references/kustomize/kustomization/

Merge configuration

Kustomize.io



kustomization.yaml

```yaml
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization
namespace: sample-app
resources:
  - web-app.yaml
  - certificate-issuer.yaml
  - identity-service.yaml
  - ingress.yaml
  - message-broker.yaml
  - namespace.yaml
  - notification-service.yaml
  - shop-service.yaml
```

web-app.yaml

```yaml
apiVersion: v1
kind: Service
```

certificate-issuer.yaml

```yaml
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
```

identity-service.yaml

```yaml
apiVersion: v1
kind: Service
```

namespace.yaml

```yaml
apiVersion: v1
kind: Namespace
```

ingress.yaml

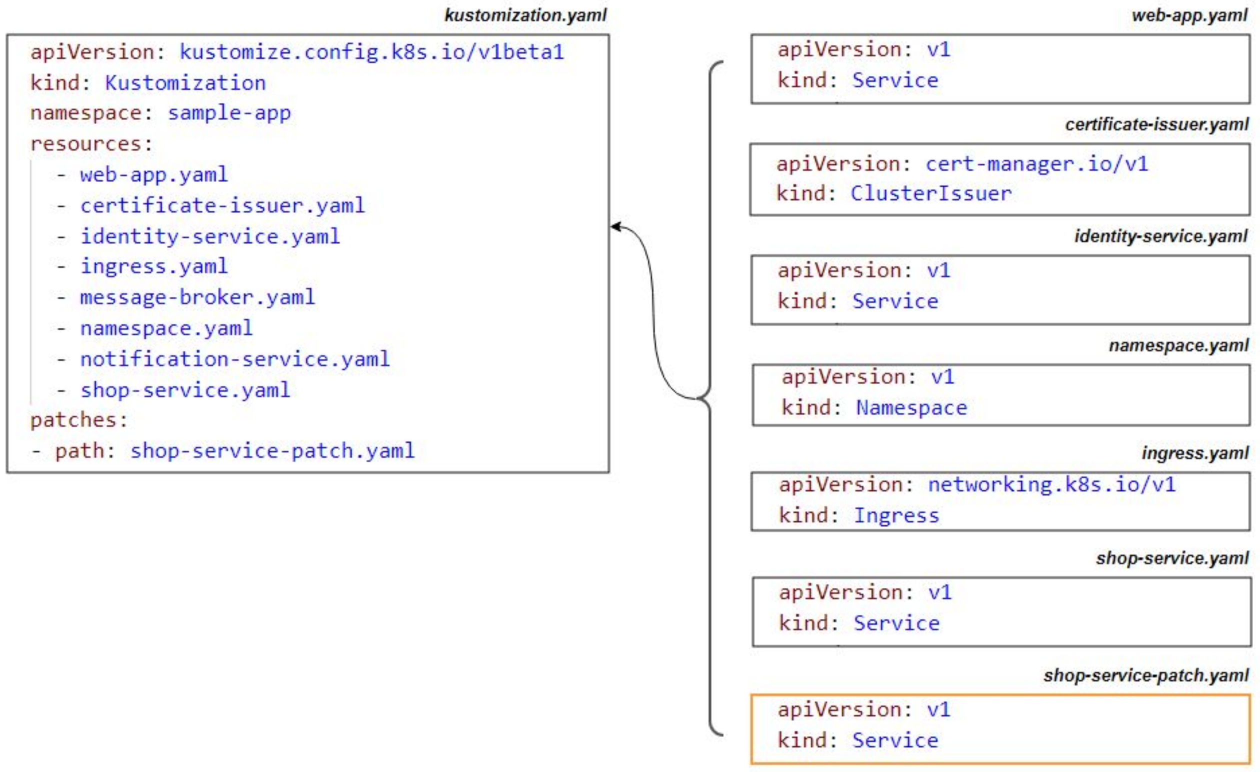
```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
```

shop-service.yaml

```yaml
apiVersion: v1
kind: Service
```

Demo

Patch configuration



kustomization.yaml

```yaml
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization
namespace: sample-app
resources:
  - web-app.yaml
  - certificate-issuer.yaml
  - identity-service.yaml
  - ingress.yaml
  - message-broker.yaml
  - namespace.yaml
  - notification-service.yaml
  - shop-service.yaml
patches:
- path: shop-service-patch.yaml
```

web-app.yaml

```yaml
apiVersion: v1
kind: Service
```

certificate-issuer.yaml

```yaml
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
```

identity-service.yaml

```yaml
apiVersion: v1
kind: Service
```

namespace.yaml

```yaml
apiVersion: v1
kind: Namespace
```

ingress.yaml

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
```

shop-service.yaml

```yaml
apiVersion: v1
kind: Service
```

shop-service-patch.yaml

```yaml
apiVersion: v1
kind: Service
```

Demo

Apply configuration

Create kubernetes configuration, build with kustomize and apply using either standalone kustomize executable or kubectl built-in feature:

```
kustomize build | kubectl apply -f -
```

or

```
kubectl apply -k ./
```

Overlay

An overlay is a kustomization that depends on another kustomization.

Overlays make the most sense when there is more than one, because they create different variants of a common base - e.g. development, QA, staging and production environment variants.

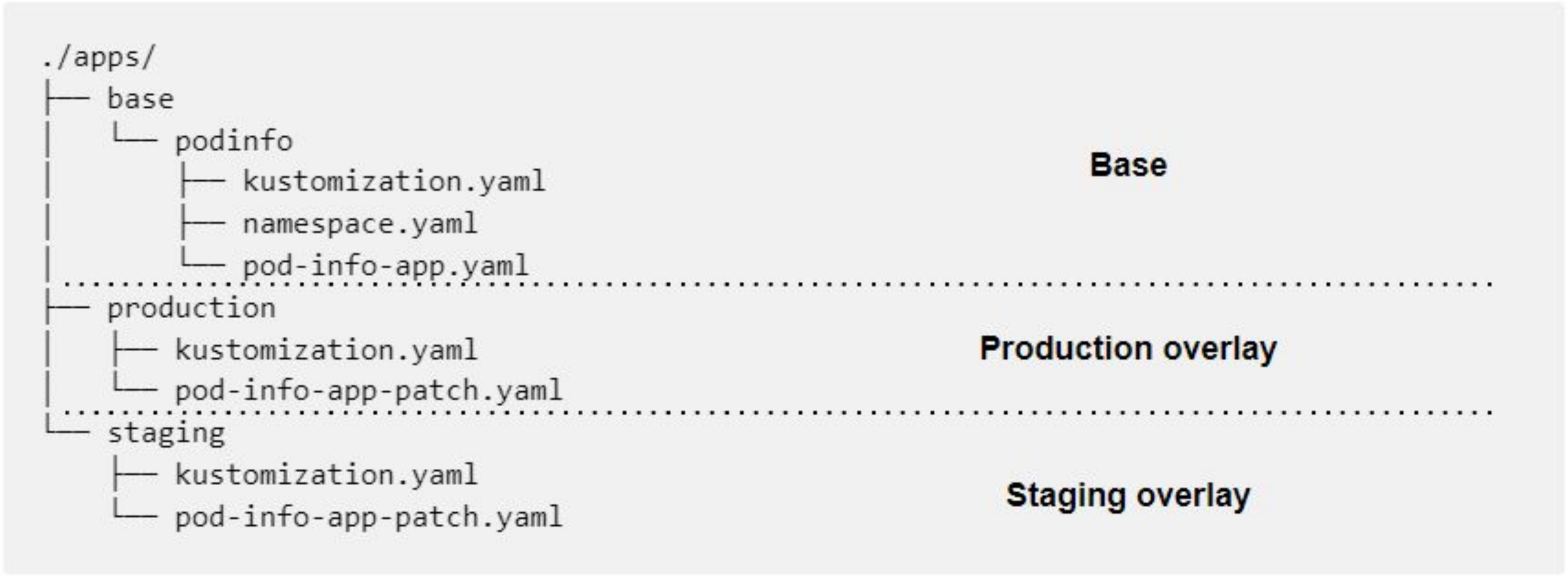
Overlay

Kustomize.io

```
./apps/
├── base
│   └── podinfo
│       ├── kustomization.yaml
│       ├── namespace.yaml
│       └── pod-info-app.yaml
├── production
│   ├── kustomization.yaml
│   └── pod-info-app-patch.yaml
└── staging
    ├── kustomization.yaml
    └── pod-info-app-patch.yaml
```

Overlay

kustomize.io



```
./apps/
├── base
│   └── podinfo
│       ├── kustomization.yaml
│       ├── namespace.yaml
│       └── pod-info-app.yaml
├── production
│   ├── kustomization.yaml
│   └── pod-info-app-patch.yaml
└── staging
    ├── kustomization.yaml
    └── pod-info-app-patch.yaml
```

Demo

Best of Kustomize

- Easy to understand & learn,
- Natively built into kubectl,
- Configured using plain YAML syntax,
- Declarative approach for configuration management,
- Integrates with FluxCD,

Introduction to FluxCD

Flux is a tool for keeping Kubernetes clusters in sync with sources of configuration (like Git repositories), and automating updates to configuration when there is new code to deploy.

Project page → https://fluxcd.io/docs/

GitHub repo → https://github.com/fluxcd/flux2

Overview

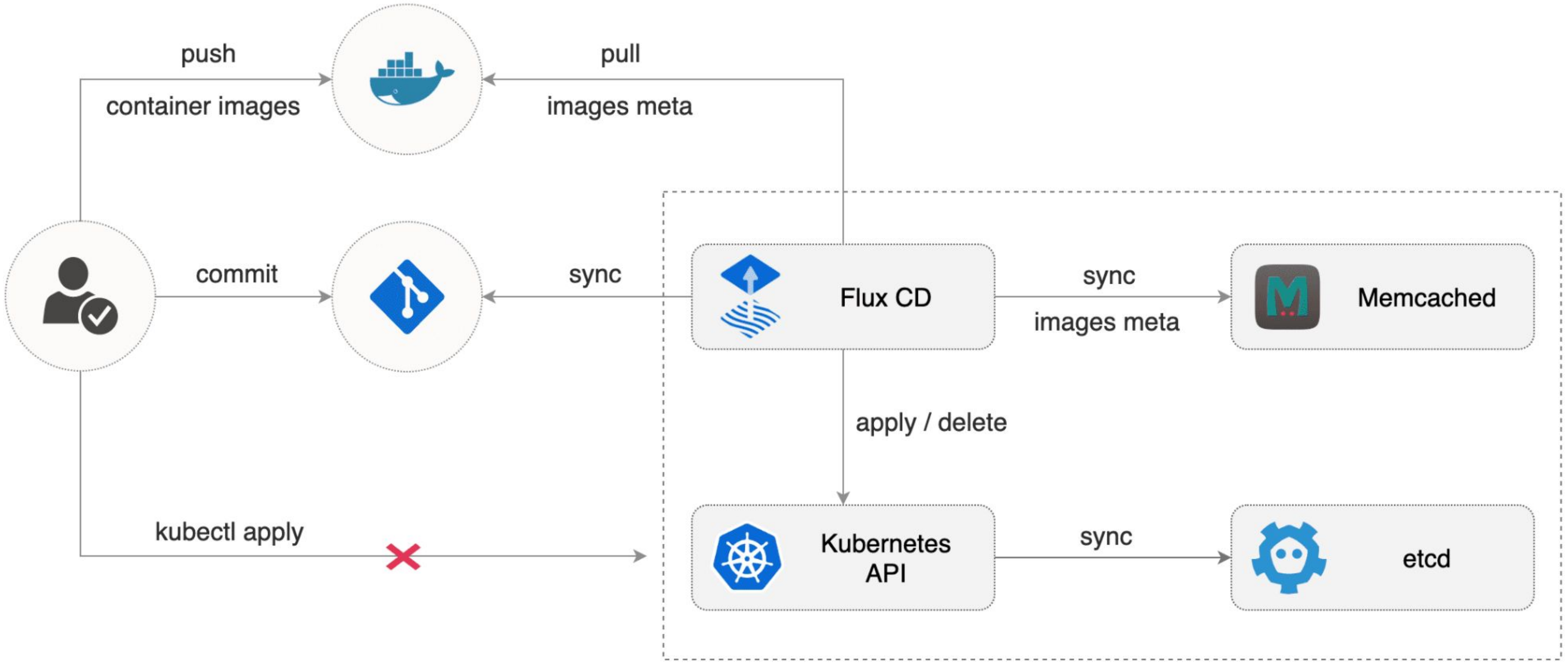


Image source: https://fluxcd.io/

Core concepts

- GitOps
- Kustomization
- Sources
- Bootstrap
- Reconciliation

Glossary: https://fluxcd.io/docs/concepts/

Install Flux CLI

Documentation https://fluxcd.io/docs/installation/#install-the-flux-cli

Releases page → https://github.com/fluxcd/flux2/releases

Based on → https://fluxcd.io/docs/get-started/

Install FluxCD on K8s

Installation process of FluxCD is called bootstrapping.

The *flux bootstrap git* command takes an existing Git repository, clones it and commits the Flux components manifests to the specified branch. Then it configures the target cluster to synchronize with that repository.

See more → https://fluxcd.io/docs/installation/#bootstrap

See more → https://fluxcd.io/docs/installation/#generic-git-server

Demo

Deploy new app version

flux

Any change to repository definition will be reflected on the cluster.

Flux periodically checks repository and apply changes.

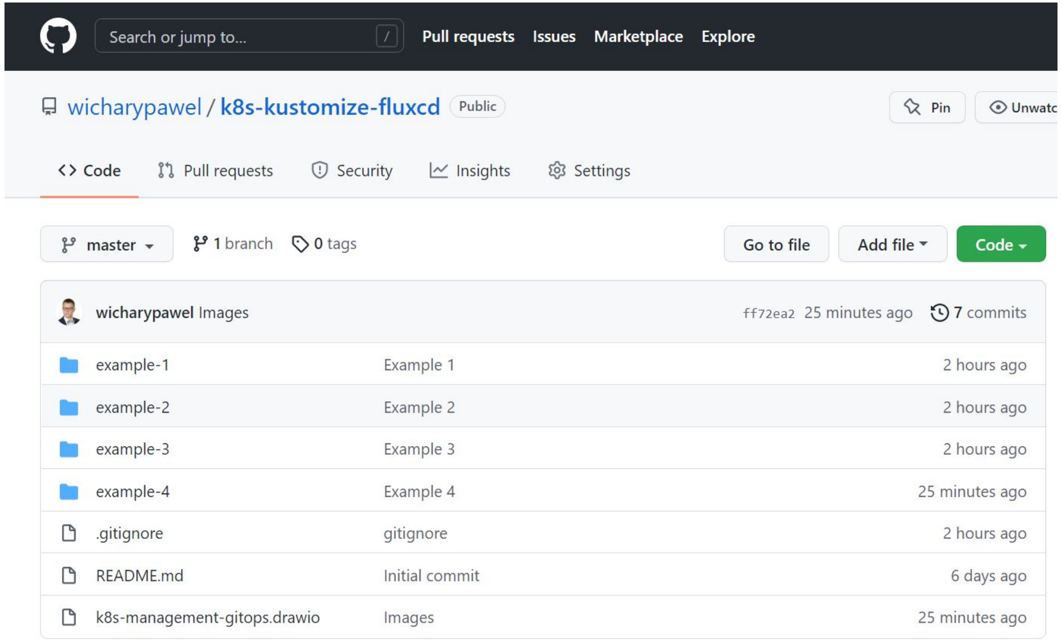
Demo

Recap

- Use Kustomize to merge and patch K8s configuration,
- Use Kustomize base and variants to avoid duplicates,
- Remember Kustomize is more than just patches,
- Use FluxCD to make your first step towards GitOps,
- Configure FluxCD using YAML files created during bootstrapping,
- Explore documentation for more,

Repository link

- https://github.com/wicharypawel/k8s-kustomize-fluxcd



wicharypawel / k8s-kustomize-fluxcd Public

1 branch 0 tags

wicharypawel Images ff72ea2 25 minutes ago 7 commits

example-1	Example 1	2 hours ago
example-2	Example 2	2 hours ago
example-3	Example 3	2 hours ago
example-4	Example 4	25 minutes ago
.gitignore	gitignore	2 hours ago
README.md	Initial commit	6 days ago
k8s-management-gitops.drawio	Images	25 minutes ago

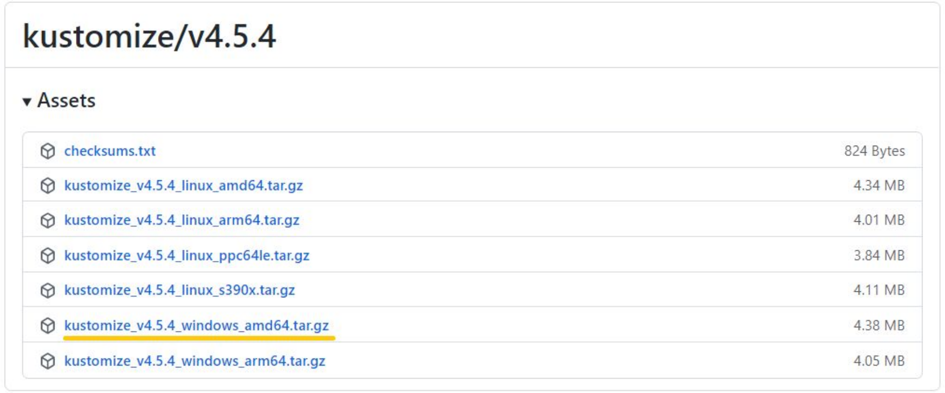
Questions

Thank you for your time

In case of questions contact me at wicharypawel@gmail.com

Extras: Install standalone Kustomize

1. Navigate to *https://kubectl.docs.kubernetes.io/installation/kustomize/*
2. Select "*Binaries*" section
3. Open "*Releases page*" (redirects to GitHub)
4. Find the latest version of kustomize
5. Download precompiled binary
6. Extract using a tool like 7-zip
7. Add executable to system PATH



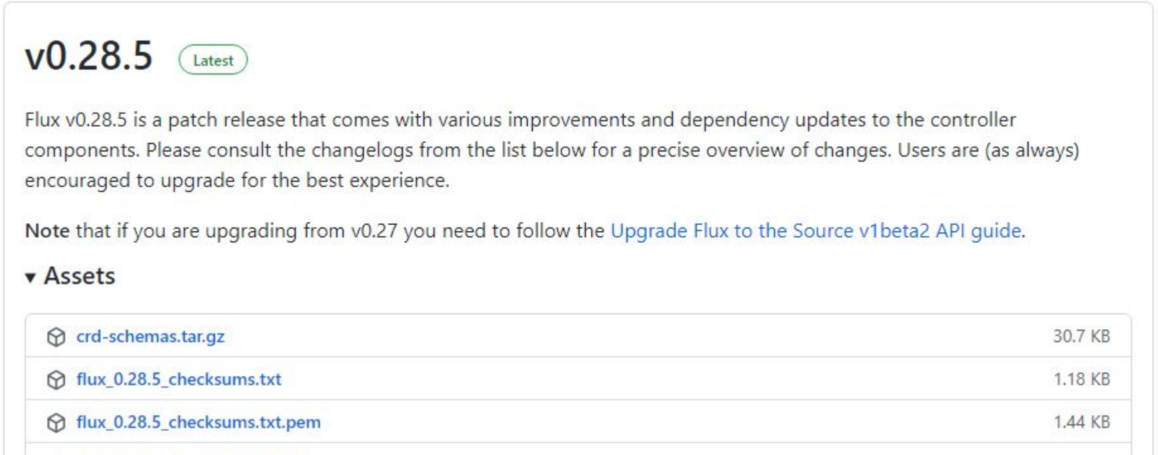
kustomize/v4.5.4

▾ Assets

checksums.txt	824 Bytes
kustomize_v4.5.4_linux_amd64.tar.gz	4.34 MB
kustomize_v4.5.4_linux_arm64.tar.gz	4.01 MB
kustomize_v4.5.4_linux_ppc64le.tar.gz	3.84 MB
kustomize_v4.5.4_linux_s390x.tar.gz	4.11 MB
kustomize_v4.5.4_windows_amd64.tar.gz	4.38 MB
kustomize_v4.5.4_windows_arm64.tar.gz	4.05 MB

Extras: Install Flux CLI

1. Navigate to *https://github.com/fluxcd/flux2/releases*
2. Find the latest version
3. Download precompiled binary
4. Extract using a tool like 7-zip
5. Add executable to system PATH



v0.28.5 Latest

Flux v0.28.5 is a patch release that comes with various improvements and dependency updates to the controller components. Please consult the changelogs from the list below for a precise overview of changes. Users are (as always) encouraged to upgrade for the best experience.

Note that if you are upgrading from v0.27 you need to follow the Upgrade Flux to the Source v1beta2 API guide.

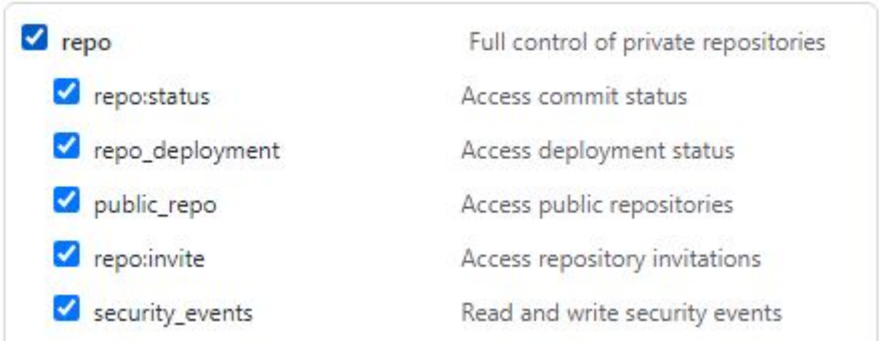
▾ Assets

crd-schemas.tar.gz	30.7 KB
flux_0.28.5_checksums.txt	1.18 KB
flux_0.28.5_checksums.txt.pem	1.44 KB

Extras: Create Personal Access Token GitHub

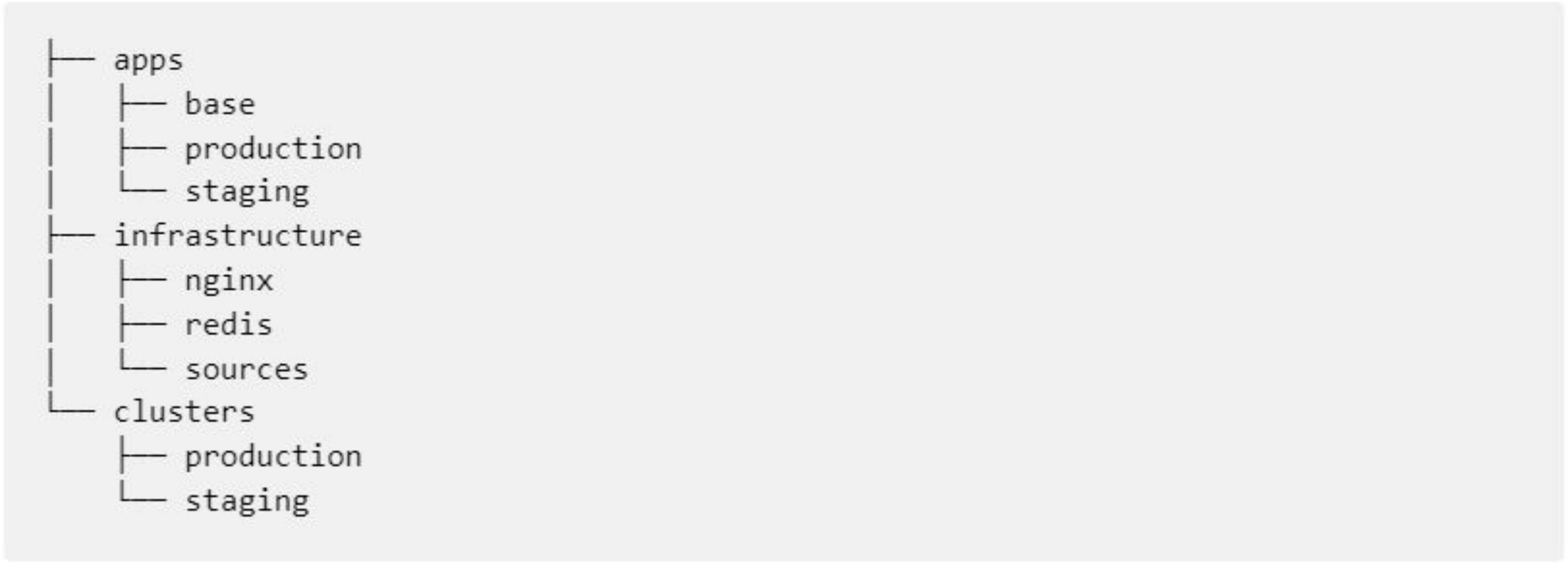
- https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token

NOTE: FluxCD needs **repo** permissions



✓ repo	Full control of private repositories
✓ repo:status	Access commit status
✓ repo_deployment	Access deployment status
✓ public_repo	Access public repositories
✓ repo:invite	Access repository invitations
✓ security_events	Read and write security events

Extras: Recommended folder structure Flux CD



```
├── apps
│   ├── base
│   ├── production
│   └── staging
├── infrastructure
│   ├── nginx
│   ├── redis
│   └── sources
└── clusters
    ├── production
    └── staging
```

Based on → https://github.com/fluxcd/flux2-kustomize-helm-example

Extras: Kustomize vs Helm

- https://defreng.medium.com/helm-vs-kustomize-how-to-deploy-your-applications-in-2020-67f4d104da69

Extras: ArgoCD vs FluxCD

- https://rajputvaibhav.medium.com/argo-cd-vs-flux-cd-right-gitops-tool-for-your-kubernetes-cluster-c71cff489d26
- https://thenewstack.io/gitops-on-kubernetes-deciding-between-argo-cd-and-flux/