


KubeCon | CloudNativeCon

North America 2021

RESILIENCE
REALIZED

Shh, It's a Secret:
Managing Your Secrets in a GitOps Way

Jacob Wernette, IBM
Josh Kayani, IBM

Jacob Wernette

KubeCon CloudNativeCon North America 2021

Site Reliability Engineer at IBM

Graduate of Central Michigan University, with a Bachelor's Degree in Computer Science.

From IBM to Red Hat and Back

- Joined IBM in January 2015
- Moved to DevOps in 2018
- Left for Red Hat in Late 2018 as a CI/CD Engineer
- Returned to IBM in February 2020 as an SRE

Technologies I am Excited About

- Kubernetes Operators/CRDs
- Argo CD or Anything GitOps
- Helm
- Terraform
- Anything Cloud Native

Josh Kayani

Site Reliability Engineer

From NC State University to IBM

- Interned at IBM in May 2018
- Graduated from North Carolina State University in December 2018
- Joined IBM in January 2019

NC STATE UNIVERSITY

IBM

Our Kubernetes Journey

Manual deployments to WebSphere of Monolith

Monolith to Microservices, became too much for team to manage

Manual resource management in Kubernetes

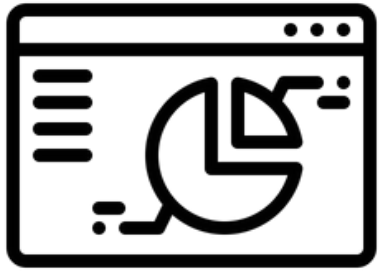
What is GitOps[1]?

1. The principle of declarative desired state
2. The principle of immutable desired state versions
3. The principle of continuous state reconciliation
4. The principle of operations through declaration

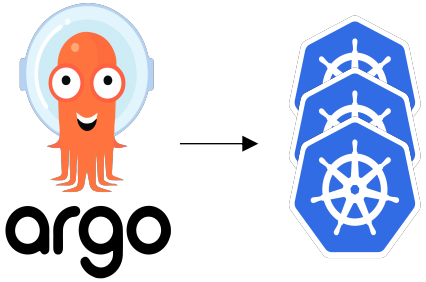
GitOps WG

[1]GitOps Working Group Principles (https://opengitops.dev/#principles)

Why Argo CD?



Argo CD User Interface



One Argo CD for Multiple Clusters

Extensibility

CNCF Incubating Project

Native Support for Helm/Kustomize/etc.

What about Secrets?

GitOps Secrets Requirements

Should be able to work with all Kubernetes Resources

Should not require a CRD or Operator

Vault credentials in one place instead of every cluster

Must be able to be used with other templating tools such as Helm

The Idea


```yaml
apiVersion: v1
kind: Secret
metadata:
  name: example-secret
data:
  sample-secret: <username>
type: Opaque
```


```yaml
apiVersion: v1
kind: Secret
metadata:
  name: example-secret
data:
  sample-secret: dXNlcm5hbWU=
type: Opaque
```

Argo CD Custom Plugins

Argo CD allows integrating more config management tools using config management plugins.

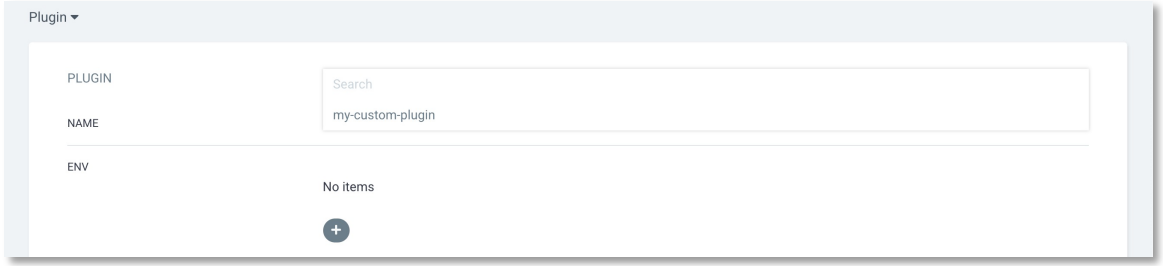
Step 1 – Make sure required binaries are available in argocd-repo-server pod via Init Container or Custom Image

```yaml
spec:
  # 1. Define an emptyDir volume which will hold the custom binaries
  volumes:
  - name: custom-tools
    emptyDir: {}
  # 2. Use an init container to download/copy custom binaries into the emptyDir
  initContainers:
  - name: download-tools
    image: alpine:3.8
    command: [sh, -c]
    args:
    - wget -qO- https://storage.googleapis.com/kubernetes-helm/helm-v2.12.3-linux-amd64.tar.gz | tar -xvzf - &&
      mv linux-amd64/helm /custom-tools/
    volumeMounts:
    - mountPath: /custom-tools
      name: custom-tools
  # 3. Volume mount the custom binary to the bin directory (overriding the existing version)
  containers:
  - name: argocd-repo-server
    volumeMounts:
    - mountPath: /usr/local/bin/helm
      name: custom-tools
      subPath: helm
```

Step 2 – Register a new plugin in the argocd-cm ConfigMap

```yaml
data:
  configManagementPlugins: |
    - name: my-custom-plugin
      init:                           # Optional command to initialize application source directory
        command: ["sample command"]
        args: ["sample args"]
      generate:                       # Command to generate manifests YAML
        command: ["sample command"]
        args: ["sample args"]
```

Step 3 – Select your plugin when creating an Application



Custom Plugin Docs: https://bit.ly/3osUhpm

Introducing argocd-vault-plugin

An Argo CD plugin to retrieve secrets from Secret Management tools and inject them into Kubernetes secrets

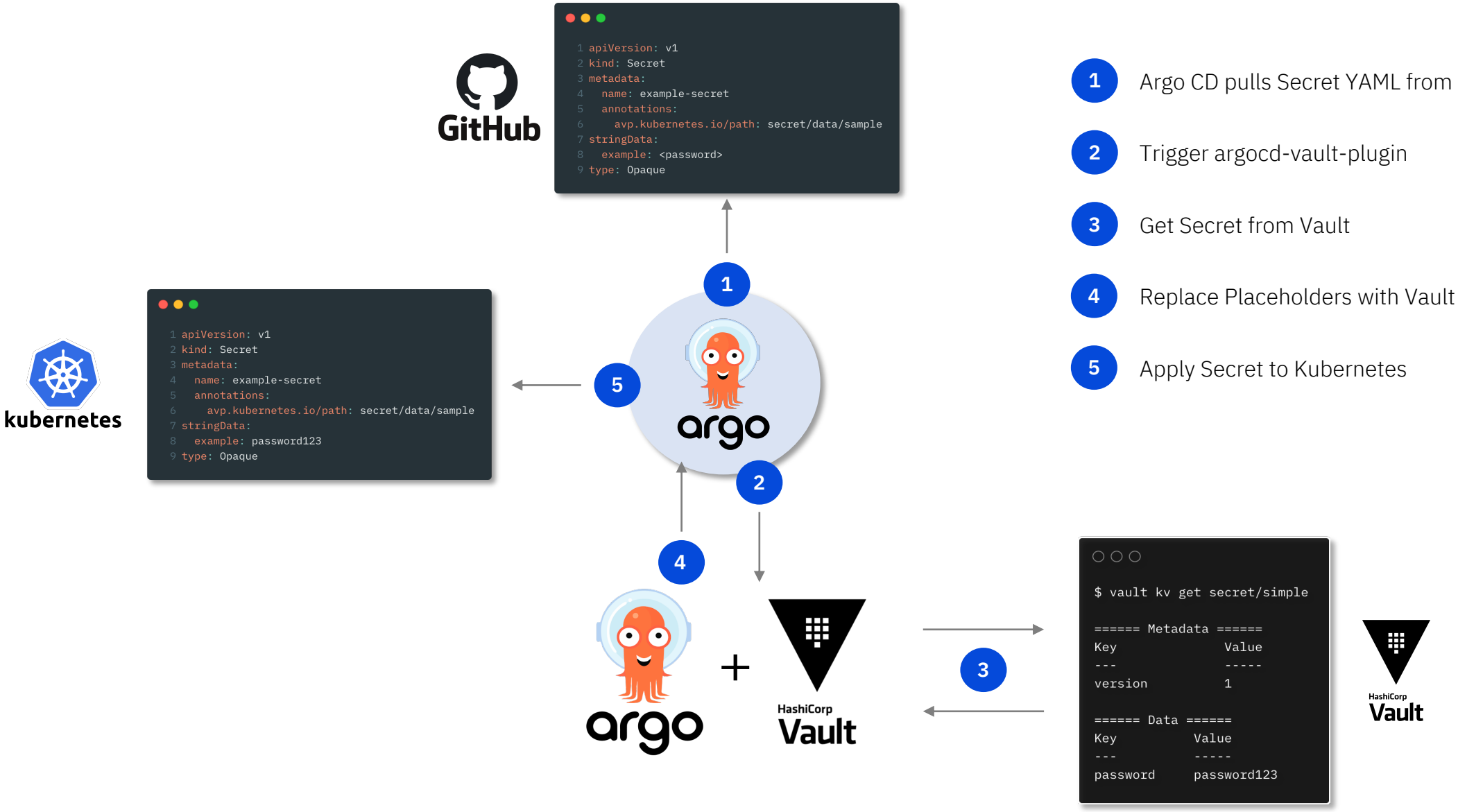
Supported Secret Managers

- HashiCorp Vault
- IBM Cloud Secrets Manager
- AWS Secrets Manager
- GCP Secrets Manager
- Azure Key Vault

Supported Features

- Secret versioning
- Annotation or inline-path based placeholders
- Base64 encoding modifier
- Can be used with Helm/Kustomize/etc.
- Multiple authentication methods
  - Vault AppRole
  - Vault K8s Auth
  - Native Secret Manager Authentication

How It Works



1. Argo CD pulls Secret YAML from GitHub
2. Trigger argocd-vault-plugin
3. Get Secret from Vault
4. Replace Placeholders with Vault Data
5. Apply Secret to Kubernetes

Demo

Resources

Repository: https://github.com/IBM/argocd-vault-plugin

Documentation: https://ibm.github.io/argocd-vault-plugin

Demo Repository : https://github.com/jkayani/avp-demo-kubecon-2021

Blogs

- Solving ArgoCD Secret Management with the argocd-vault-plugin
- Introducing argocd-vault-plugin v1.0!
- How to Use HashiCorp Vault and Argo CD for GitOps on OpenShift

RESILIENCE
REALIZED

KubeCon | CloudNativeCon

North America 2021

Thank You!