HashiCorp Vault

Vault Kubernetes Integration

February 2023

Agenda

1. Helm Chart for Vault
2. Pod Secret Access
3. Vault Agent Injector
4. Container Storage Interface
5. Resources

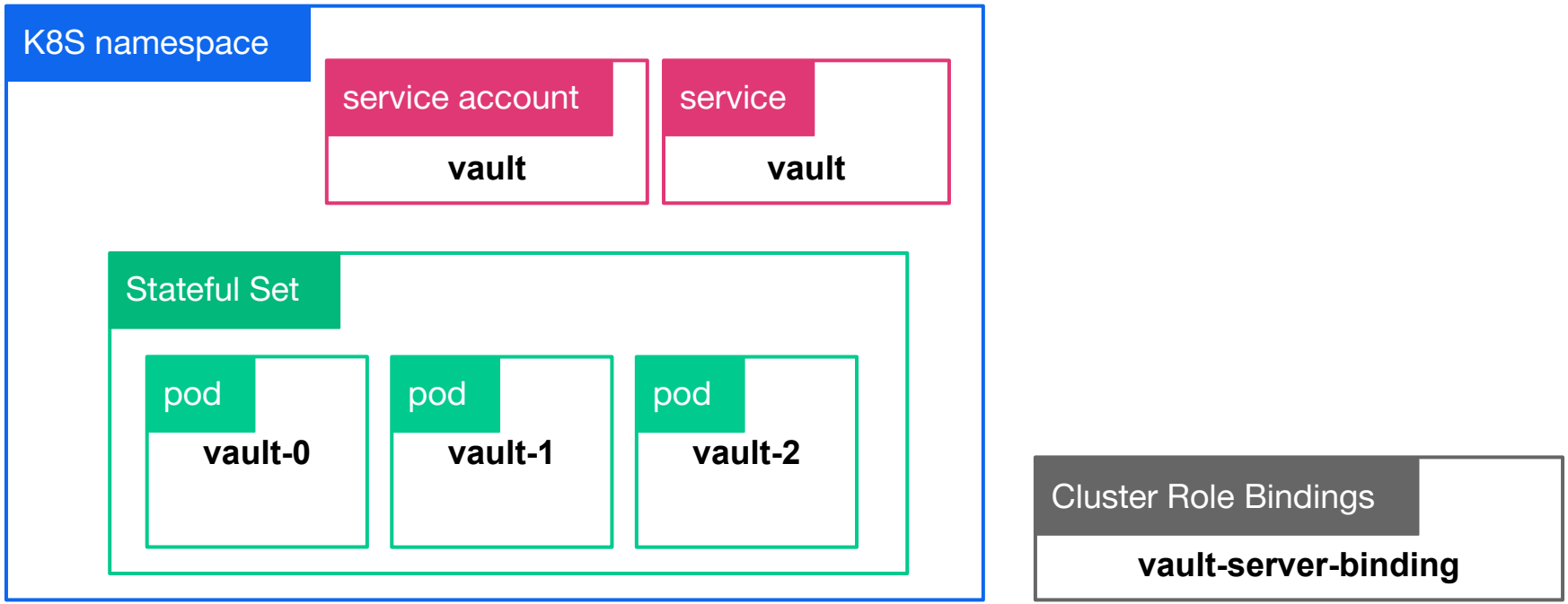
01

Helm Chart for Vault

Helm Chart for Vault

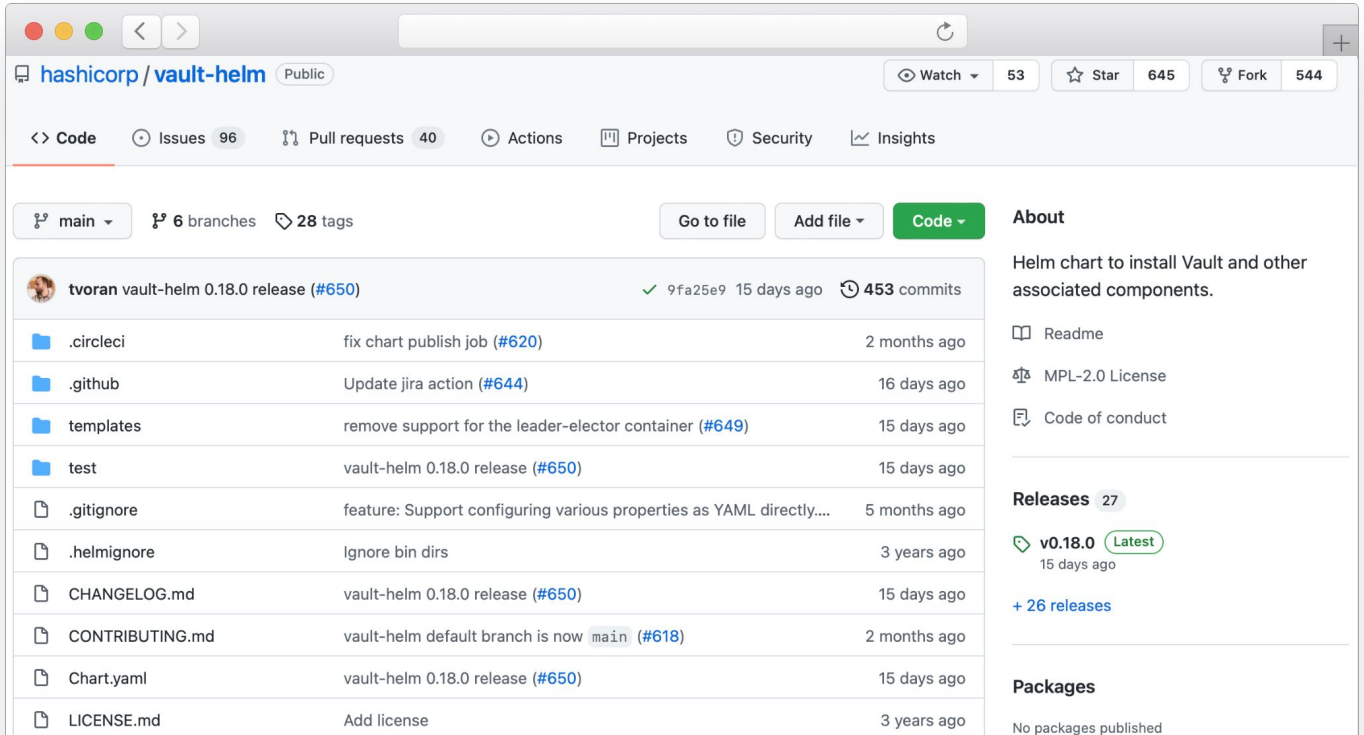
- Deployment via Helm is the recommended installation and configuration method for Vault on Kubernetes
- The Helm chart can be used to install a Vault server cluster and/or the Agent Injector
- Managing your Vault deployment using Helm can also simplify lifecycle management of your Vault Servers
- Vault Helm chart is compatible with Helm 3.6+ and Kubernetes 1.16+

Vault in Kubernetes



Vault Helm Chart

[hashicorp/vault-helm](https://github.com/hashicorp/vault-helm)

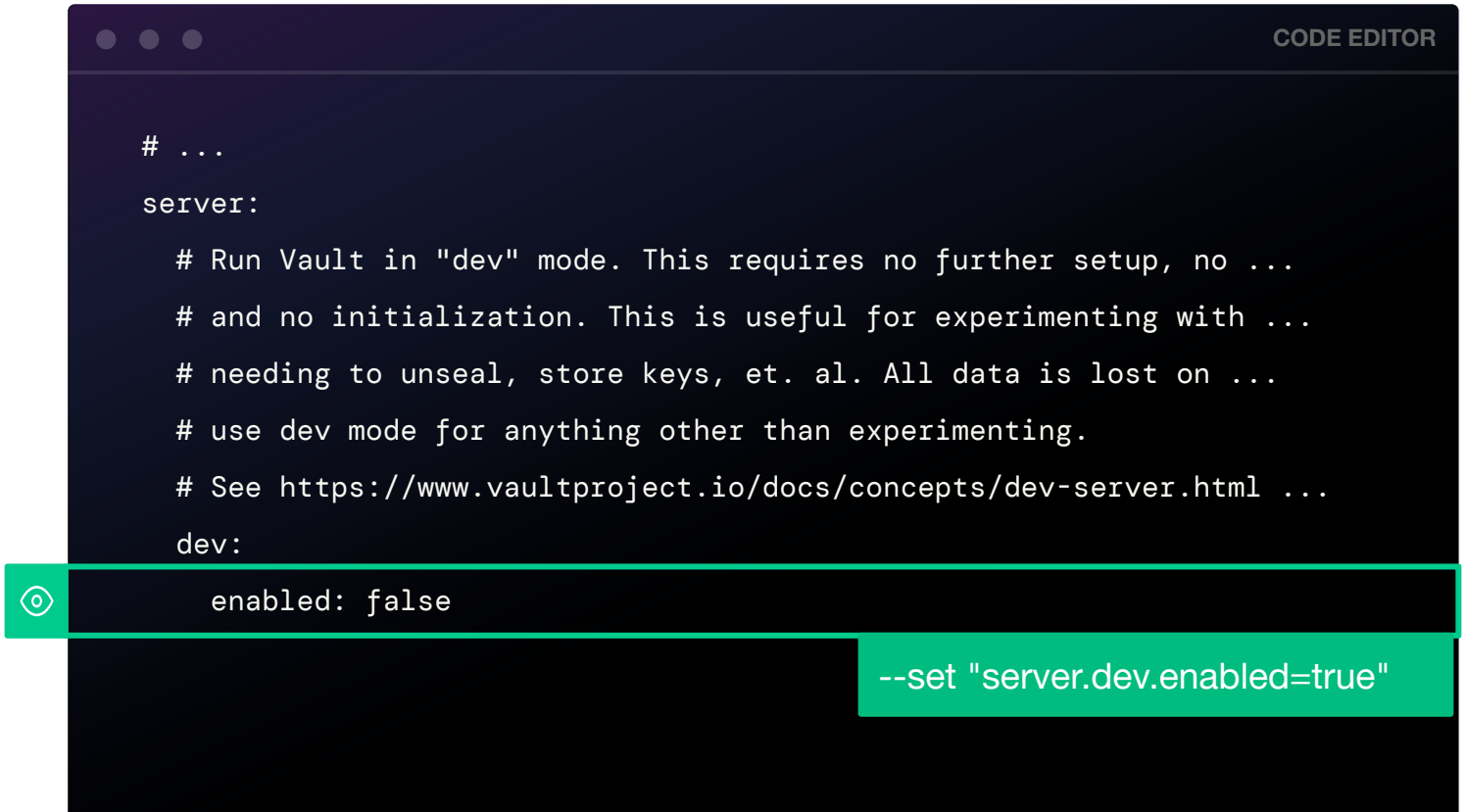


Helm Repository

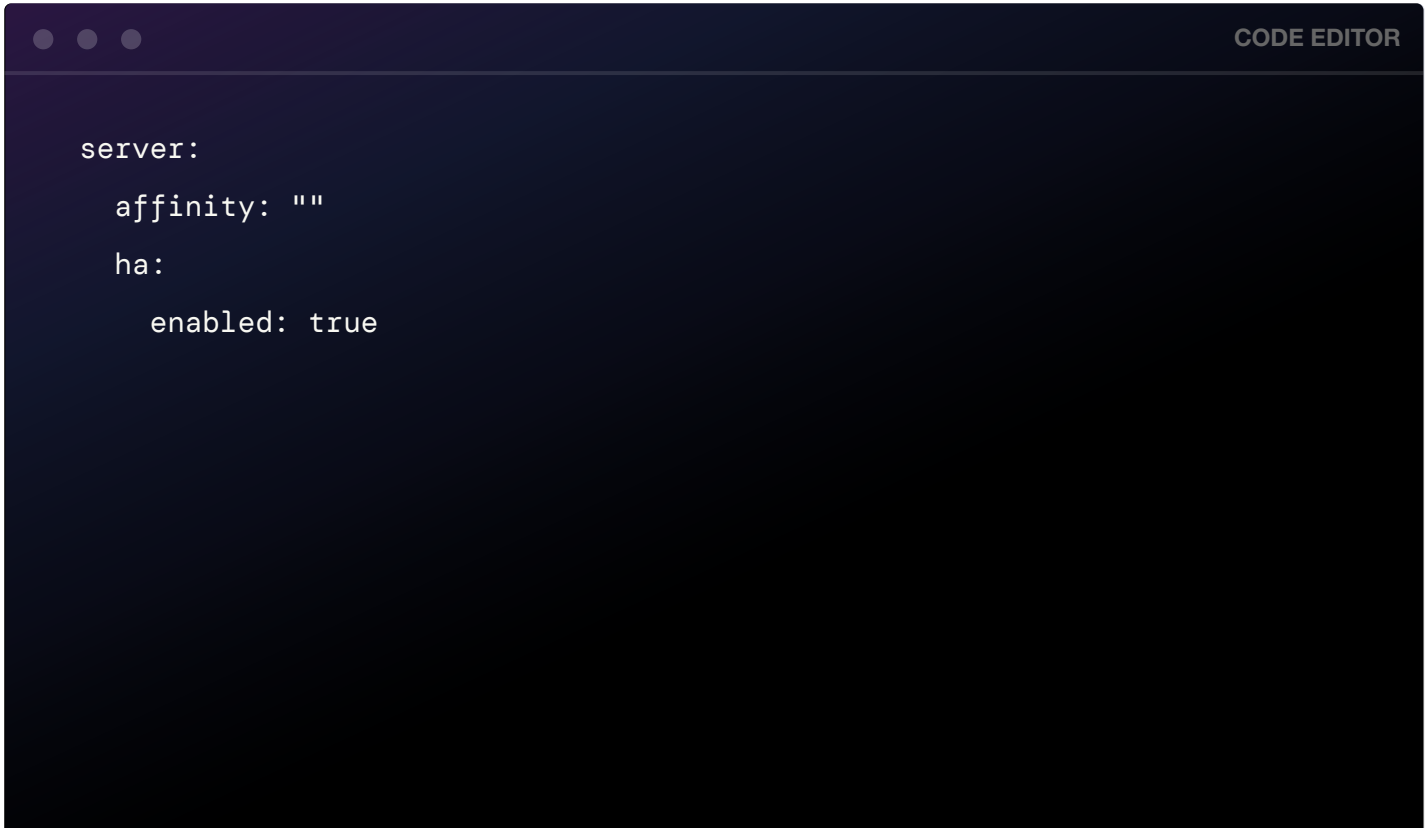
TERMINAL

```
> helm repo add hashicorp \
https://helm.releases.hashicorp.com
“Hashicorp” has been added to your repositories

> helm search repo
hashicorp/consul ...
hashicorp/vault …

> helm install vault hashicorp/vault
NAME: vault
...
```

Default Values



Override Default Values in a File



Licensing

TERMINAL

```
> secret=$(cat licensefile.hclic)

> kubectl create secret generic vault-ent-license
--from-literal="license=${secret}"

> helm install hashicorp hashicorp/vault -f config.yaml

> kubectl exec -ti vault-0 -- vault license get
```

Licensing

CODE EDITOR

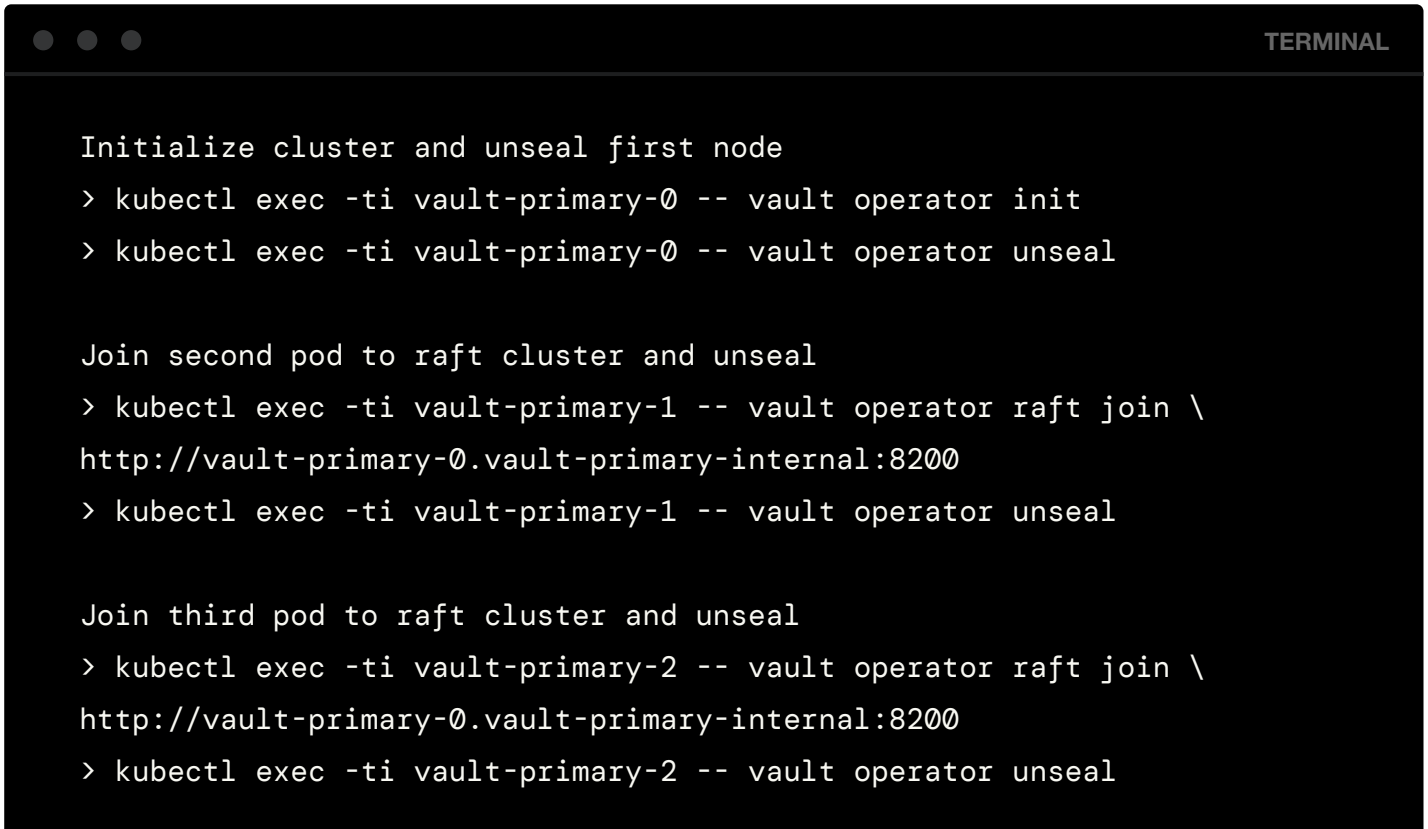
```
# config.yaml
server:
  image:
    repository: hashicorp/vault-enterprise
    tag: 1.9.0_ent
  enterpriseLicense:
    secretName: vault-ent-license
```

Primary HA Vault ENT Cluster Deployment

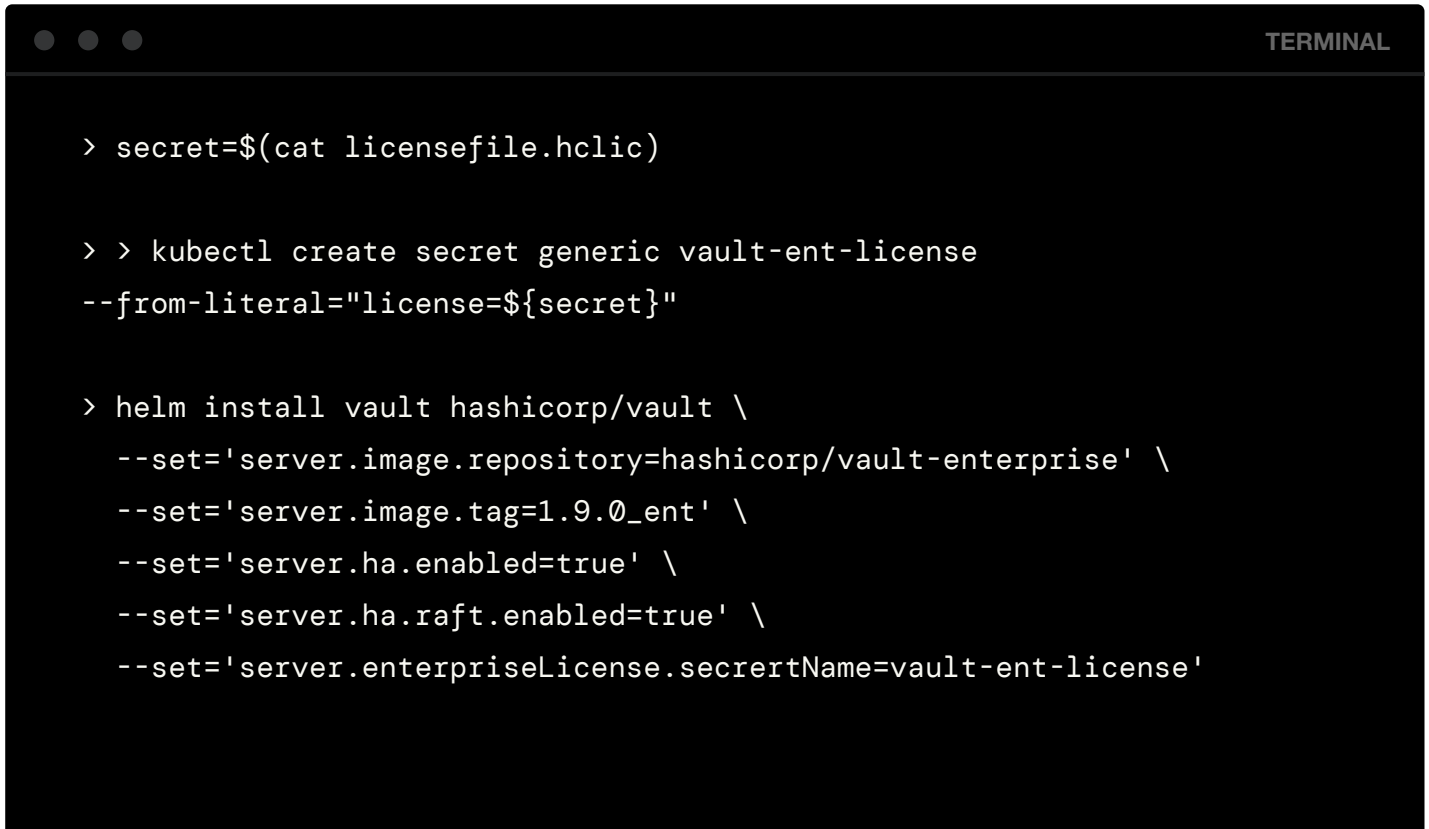
TERMINAL

```
> secret=$(cat licensefile.hclic)

> > kubectl create secret generic vault-ent-license
--from-literal="license=${secret}"

> helm install vault hashicorp/vault \
  --set='server.image.repository=hashicorp/vault-enterprise' \
  --set='server.image.tag=1.9.0_ent' \
  --set='server.ha.enabled=true' \
  --set='server.ha.raft.enabled=true' \
  --set='server.enterpriseLicense.secrertName=vault-ent-license'
```

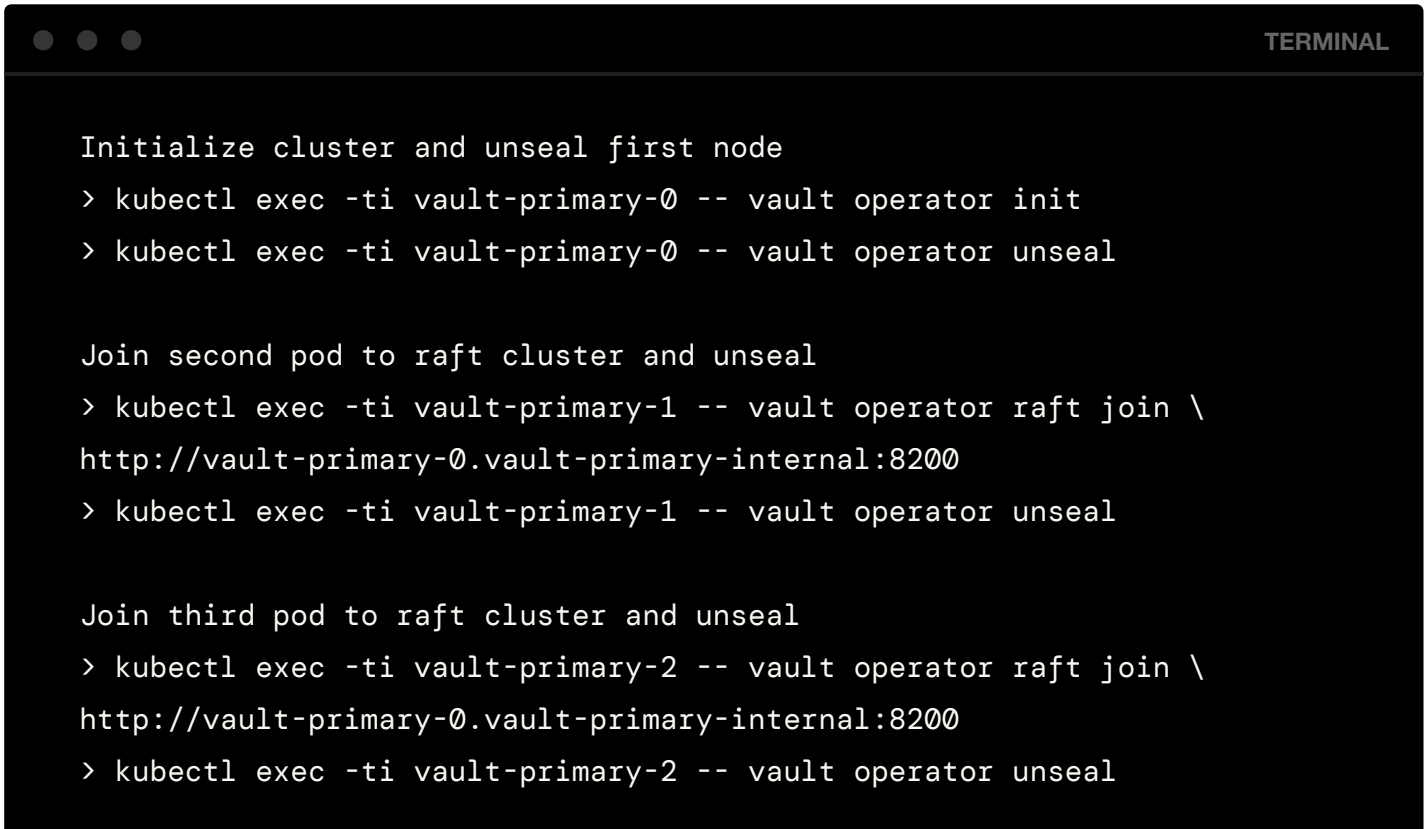
Primary HA Vault ENT Cluster Deployment



DR HA Vault ENT Cluster Deployment



DR HA Vault ENT Cluster Deployment



Enable Disaster Recovery Replication

Primary Cluster

TERMINAL

```
> kubectl exec -ti vault-primary-0 -- vault write -f
sys/replication/dr/primary/enable
primary_cluster_addr=https://vault-primary-active:8201

> kubectl exec -ti vault-primary-0 -- vault write
sys/replication/dr/primary/secondary-token id=secondary
```

Enable Disaster Recovery Replication

Secondary Cluster

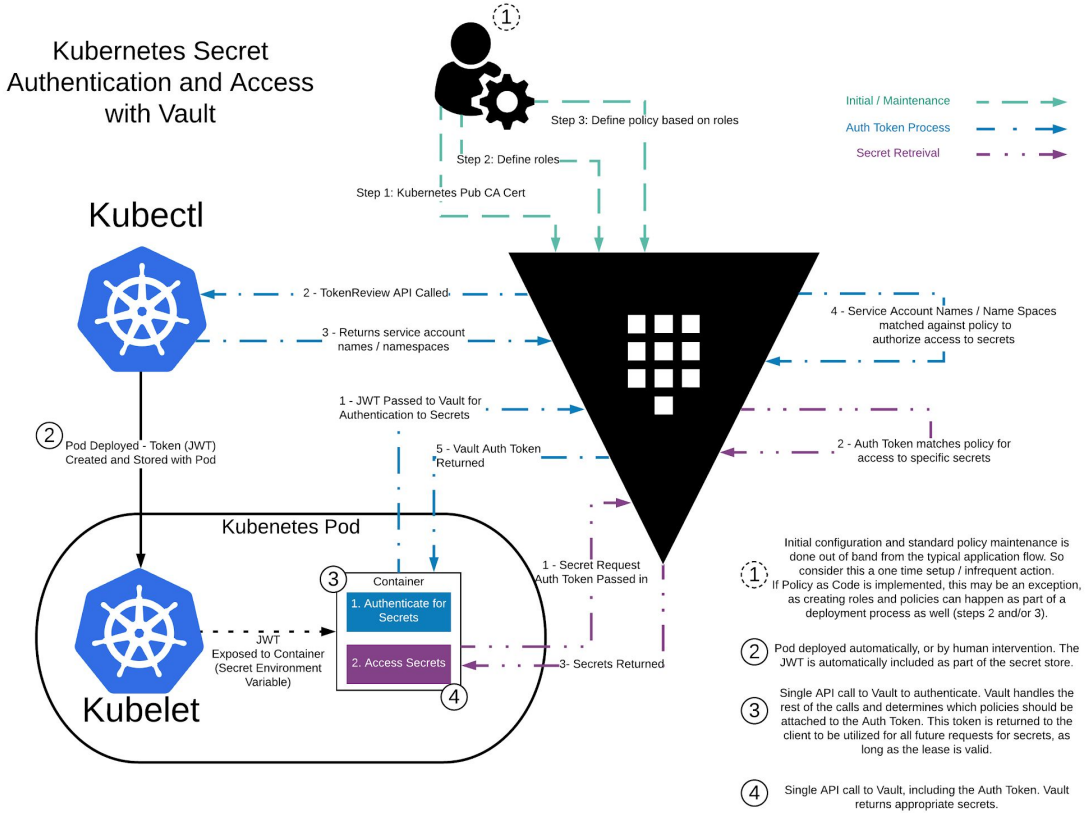
TERMINAL

```
> kubectl exec -ti vault-secondary-0 -- vault write
sys/replication/dr/secondary/enable token=<TOKEN FROM
PRIMARY>

> kubectl delete pod vault-secondary-1
> kubectl exec -ti vault-secondary-1 -- vault operator
unseal <PRIMARY UNSEAL TOKEN>

> kubectl delete pod vault-secondary-2
> kubectl exec -ti vault-secondary-2 -- vault operator
unseal <PRIMARY UNSEAL TOKEN>
```

02

Pod Secret Access

Kubernetes Auth Flow



Application Pod Definition

CODE EDITOR

```
apiVersion: v1
kind: Pod
...
spec:
  serviceAccountName: k8s-service-acct
  containers:
    - name: app
      image: burtlo/exampleapp-ruby:k8s
       env:
        - name: VAULT_ADDR
        - value:
"http://vault.default.svc.cluster.local:8200"
        - name: VAULT_ROLE
        - value: "internal-app"
```

Example App Code Changes

CODE EDITOR

```
response =
HTTP.put("#{vault_url}/v1/auth/kubernetes/login") do
|req|
  req.headers['Content-Type'] = 'application/json'
  req.body = { "role" => vault_role, "jwt" => jwt
}.to_json
end

vault_token =
JSON.parse(response.body)["auth"]["client_token"]

logger.info "Received Vault Token: [#{vault_token}]"
```

03

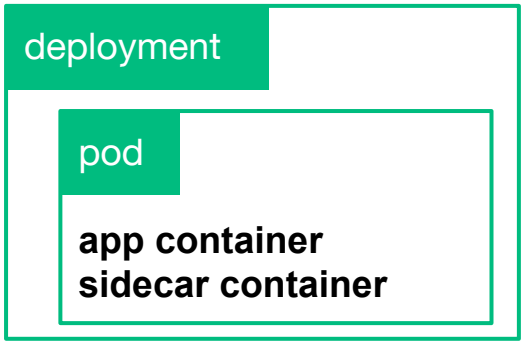
Vault Agent Injector

Sidecar Pattern

Vault unaware pods would offload the authentication and secret retrieval to a dedicated container appended to every deployment/pod

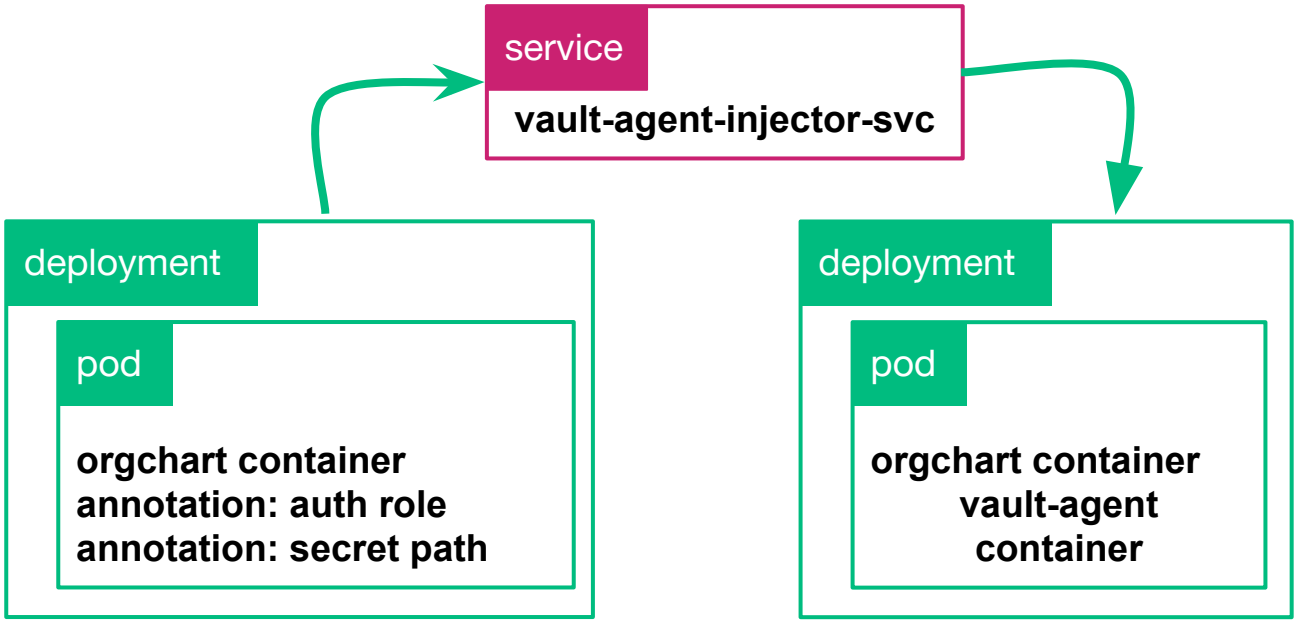
Sidecar container needs:

- Vault address
- Vault authentication role
- Vault secret path



Sidecar Pattern

Registers a Mutating Webhook Configuration that takes action when pod/deployment annotations are defined

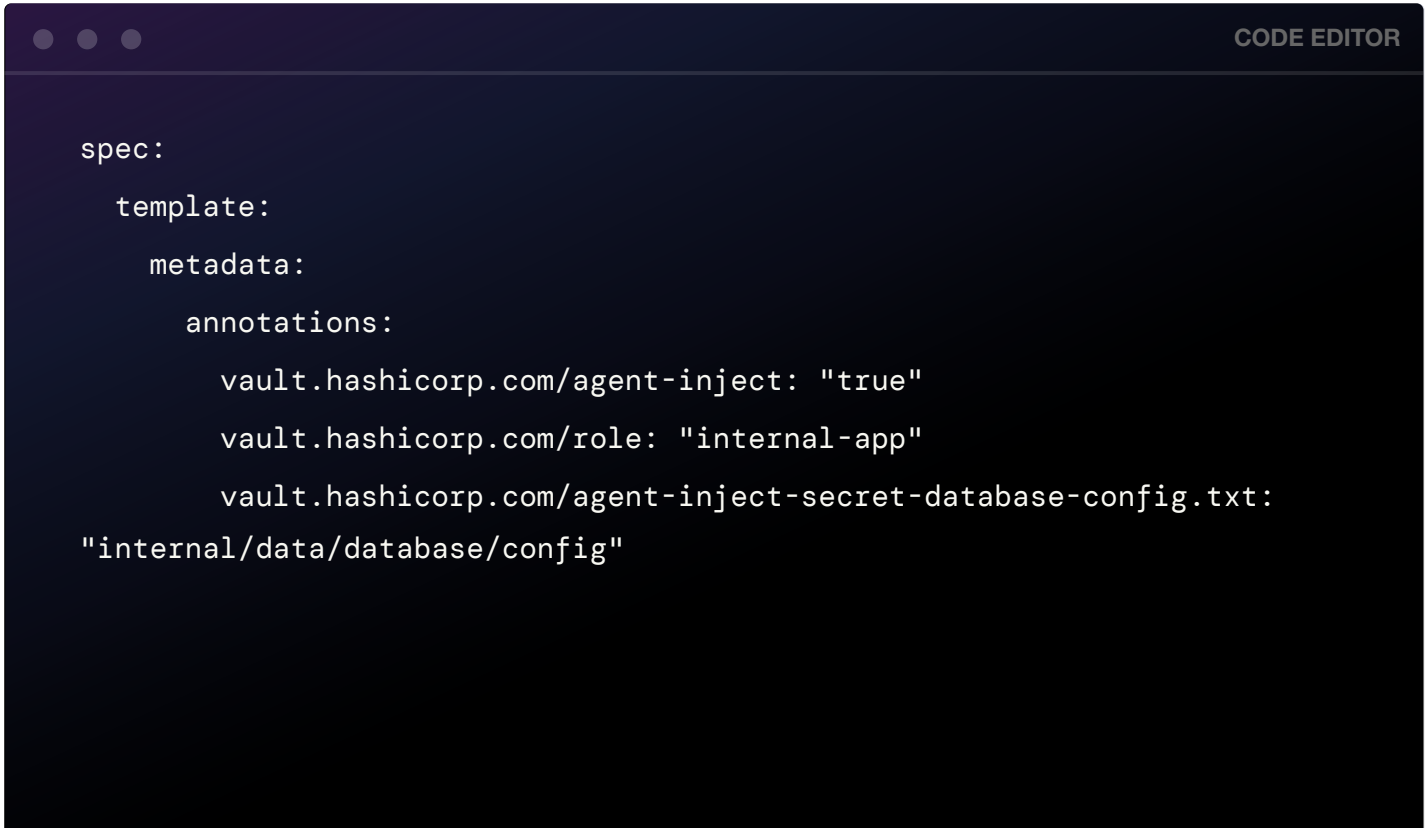


Install Agent Injector

TERMINAL

```
> helm repo add hashicorp
https://helm.releases.hashicorp.com
"hashicorp" has been added to your repositories

> helm search repo hashicorp/vault
NAME            CHART VERSION   APP VERSION DESCRIPTION
hashicorp/vault 0.18.0          1.9.0       Official
HashiCorp Vault Chart

> helm install vault hashicorp/vault \
--set="injector.enabled=true"
```

Agent Annotations



View the Secret

TERMINAL

```
> kubectl exec orgchart --container orgchart \
    -- cat /vault/secrets/database-config.txt

data: map[password:db-secret-password
username:db-readonly-user]
metadata:
map[created_time:2019-12-20T18:17:50.930264759Z
deletion_time: destroyed:false version:2]
```

04

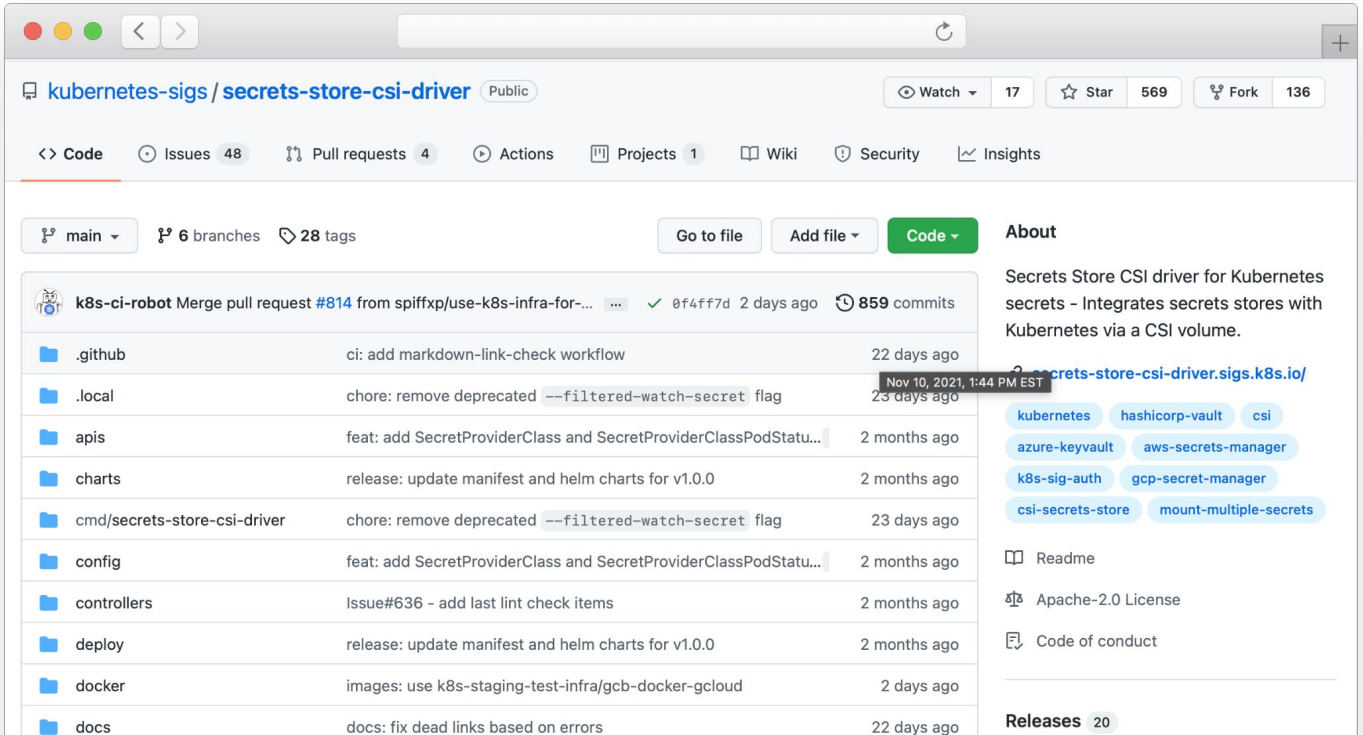
Container Storage Interface

Overview

- Secrets Store CSI driver for Kubernetes secrets - Integrates secrets stores with Kubernetes via a Container Storage Interface (CSI) volume
- The Secrets Store CSI driver allows Kubernetes to mount multiple secrets, keys, and certs stored in enterprise-grade external secrets stores into their pods as a volume
- Once the Volume is attached, the data is mounted into the container's file system

Secrets Store CSI Driver

CSI Driver



Install Container Storage Interface

TERMINAL

```
> helm repo add hashicorp
https://helm.releases.hashicorp.com
"hashicorp" has been added to your repositories

> helm search repo hashicorp/vault
NAME            CHART VERSION   APP VERSION DESCRIPTION
hashicorp/vault 0.18.0          1.9.0       Official
HashiCorp Vault Chart

> helm install vault hashicorp/vault \
 --set "injector.enabled=false" \
 --set "csi.enabled=true" \
 --set "injector.externalVaultAddr=http://addr:8200"
```

Install Secrets Store CSI Driver

TERMINAL

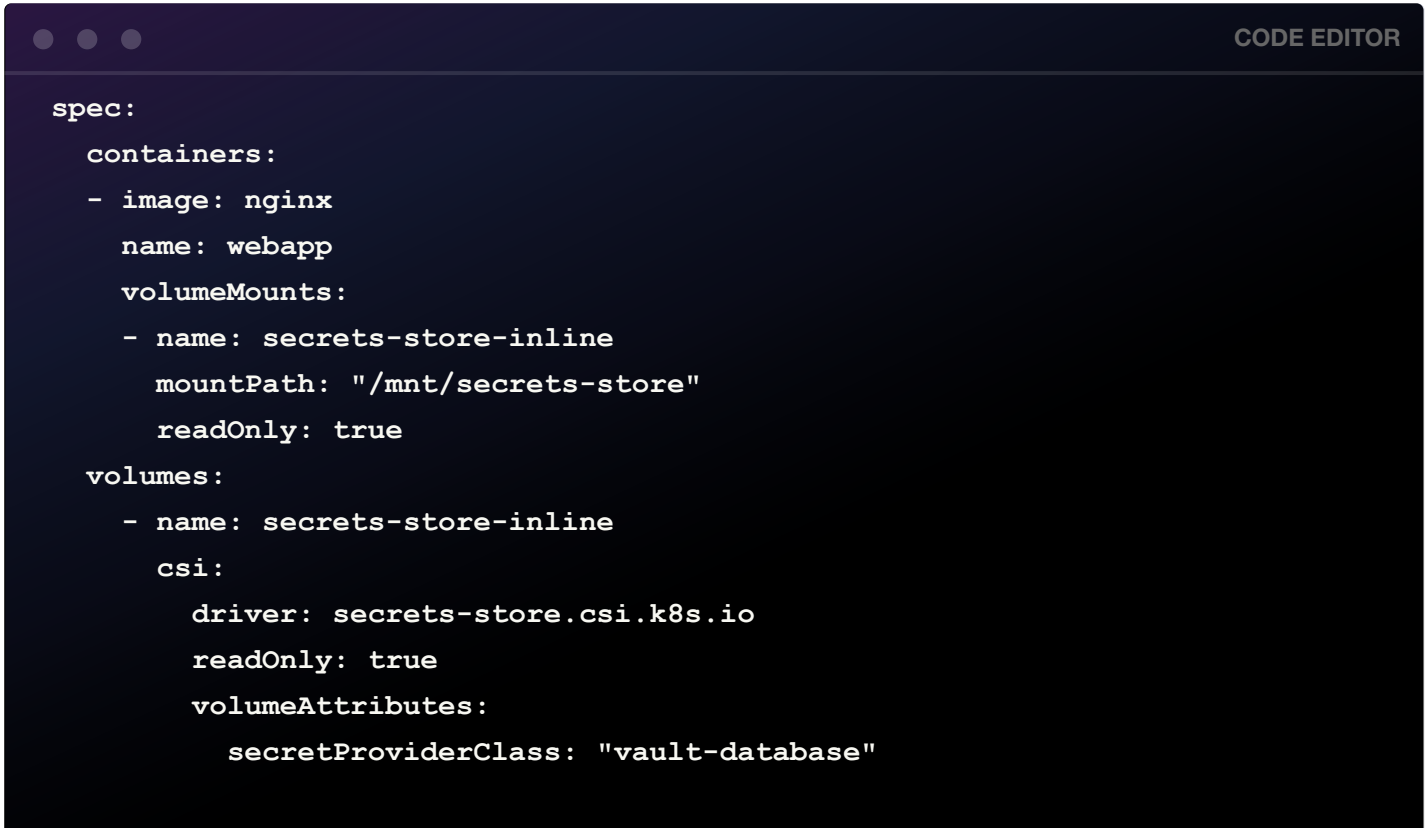
```
> helm repo add secrets-store-csi-driver \
https://raw.githubusercontent.com/kubernetes-sigs/secrets-store-csi-driver/master/charts
...

> helm install csi
secrets-store-csi-driver/secrets-store-csi-driver
...
```

Define SecretProviderClass

CODE EDITOR

```
apiVersion: secrets-store.csi.x-k8s.io/v1alpha1
kind: SecretProviderClass
metadata:
 name: vault-database
spec:
  provider: vault
  parameters:
    vaultAddress: "http://vault.default.svc.cluster.local:8200"
    roleName: "internal-app"
    objects: |
      - objectName: "db-password"
        secretPath: "internal/data/database/config"
        secretKey: "password"
```

Define a Pod with a Volume



Pattern Comparison

[Kubernetes Vault Integration via Sidecar Agent Injector vs. CSI Provider]

	Agent Sidecar	CSI
Secret projection	Shared Memory Volume Environment Variable	Ephemeral Disk Environment Variables Kubernetes Secrets
Secret scope	Global	Global
Secret types	All Secret Engines (Static & Dynamic)	All Secret Engines (Static & Dynamic)
Secret templating	Yes	No
Secret size limit	No Limit (both storage types)	No Limit (both storage types)
Secret definitions	CLI / API / UI	CLI / API / UI
Encryption	Yes (at rest & in-transit)	Yes (at rest & in-transit)
Secret rotation	Yes	No
Secret caching	Yes	No
Auditability	Yes	Yes
Deployment method	1 Shared K8s Cluster Service + 1 Sidecar Container Per Application Pod	Daemonset
Vault agent support	Yes	No
Helm support	Yes	Yes

05

Resources

Resources

- Vault on Kubernetes Security Considerations
- Vault on Kubernetes Reference Architecture
- Vault Helm Chart
- Vault Enterprise License Management - Kubernetes
- Helm Chart Examples
- Running Vault - OpenShift
- Tutorials - Vault Installation to Managed Kubernetes Services
  - Google GKE
  - Azure AKS
  - Amazon EKS
- Injecting Secrets into Kubernetes Pods via Vault Agent Containers
- Mount Vault Secrets through Container Storage Interface (CSI) Volume
- Integrate a Kubernetes Cluster with an External Vault

Q & A

Thank You

customer.success@hashicorp.com
www.hashicorp.com