

Injecting Secrets into Kubernetes Pods via Vault Agent Containers

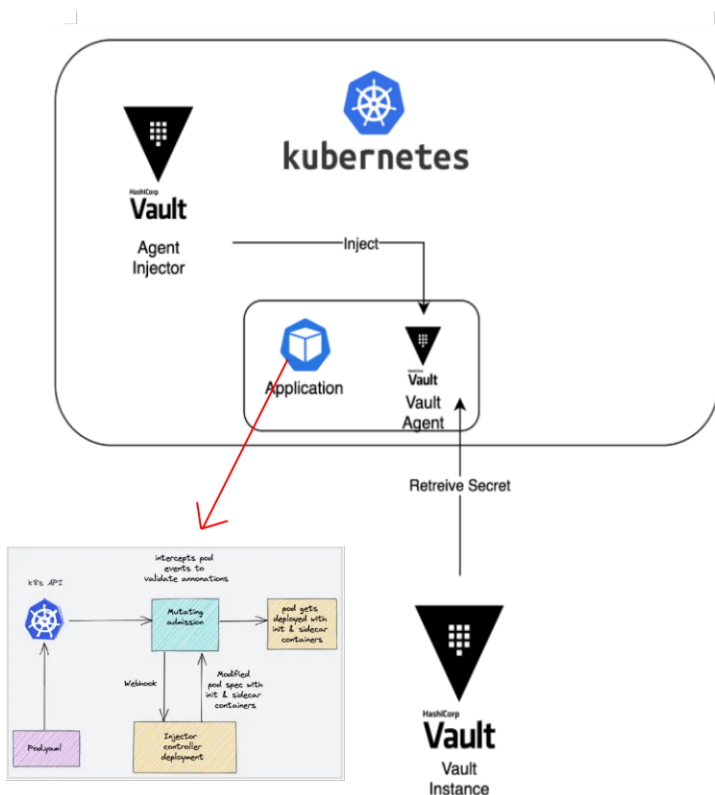
R&D

Exported on 04/19/2023

Table of Contents

No headings included in this document

- Ở đây sẽ triển khai một application lên kubernetes cluster với các environment được lấy secret trực tiếp từ một external vault: <https://vault.api-connect.io/>
- Yêu cầu:
 - Kiến thức:
 - Helm v3
 - Kubernetes, memory volume on k8s and kubernetes admission
 - Vault policy
 - Tools:
 - Helm CLI
 - Kubectl
 - Vault CLI
 - Infra:
 - Kubernetes cluster
 - Vault server



1. What is Vault Agent Injector

- Vault Agent injector được sử dụng để connect tới vault server để lấy về vault secret và đưa nó vào một shared memory volume sử dụng [Vault Agent Template](https://www.vaultproject.io/docs/agent/template)¹. Bằng cách lấy secret về và đưa vào shared memory volume, các container bên trong 1 pod có thể sử dụng Vault secret này mà không cần kết nối tới Vault Server.
- Injector là một [Kubernetes Mutation Webhook Controller](https://kubernetes.io/docs/reference/access-authn-authz/admission-controllers/)². Đây là 1 tính năng của k8s cho phép bạn có thể customize những gì được phép run trên cluster. Nếu Annotation tồn tại bên trong request thì controller này

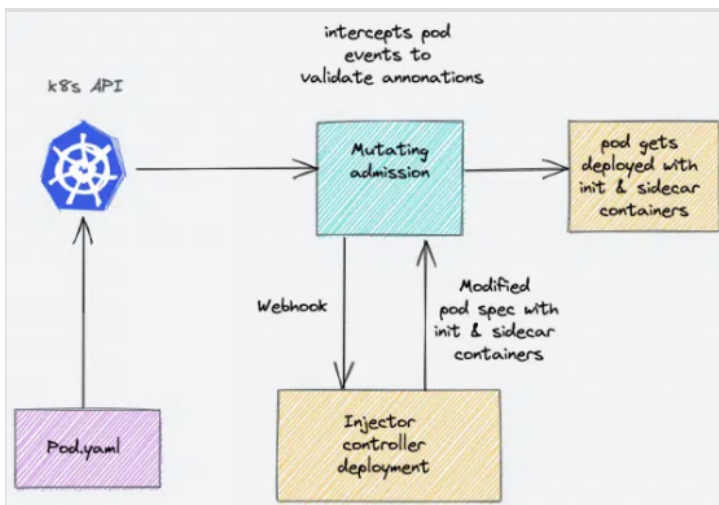
¹ <https://www.vaultproject.io/docs/agent/template>

² <https://kubernetes.io/docs/reference/access-authn-authz/admission-controllers/>

sẽ chặn lại các event trong pod và đưa vào các customize config tới pod. Chức năng này được cung cấp bởi project [vault-k8s](https://github.com/hashicorp/vault-k8s)³ vào được tự động cài đặt và cấu hình khi sử dụng Vault Helm chart

- How Vault Agent Injector work:

- Vault Agent injector làm việc bằng cách chặn các event "CREATE" và "UPDATE" trong k8s. "Controller" sẽ phân tích cú pháp và tìm kiếm thông tin trong Annotation: vault.hashicorp.com/agent-inject⁴: **true**. Nếu tìm thấy, "Controller" sẽ thay thế "spec" của pod dựa trên các annotation hiện có.
- Custom resource "MutatingWebhookConfiguration" send một webhook với thông tin của các pod để deploy một Injector controller
- Controller sẽ thêm vào Pod Spec một sidecar và một init container.
- Controller trả lại các Object đã được modified trở lại admission webhook để validate
- Sau khi validate, Pod spec được deploy với một sidecar và init container. Như vậy, khi Pod running, nó sẽ có application container, sidecar và init container



1. Intergrate Kubernetes cluster with External Vault

kubectl create serviceaccount interner-app

kubectl create ns dungla

- Deloy vault agent injector

```
helm repo add hashicorp https://helm.releases.hashicorp.com
```

```
helm repo update
```

```
helm install vault hashicorp/vault --set "injector.externalVaultAddr=https://vault.api-connect.io/" --namespace [namespace_name]
```

- Create Vault Secret & policy

- Enable the vault kv engine (key-value store).

- vault secrets enable -version=2 -address=\${VAULT_ADDR} -path env kv**

- Create secret

- vault kv put -address=\${VAULT_ADDR} env/my-secret username=foobazuser password=foobazpass**

³ <https://github.com/hashicorp/vault-k8s>

⁴ <http://vault.hashicorp.com/agent-inject>

- Create vault policy

- **vault policy write svc-policy - <<EOF**

```
path "env/data/secret" {

  capabilities = ["read"]

}

EOF
```

- Enable Kubernetes authentication.

- - vault auth enable kubernetes
 - VAULT_HELM_SECRET_NAME=\$(kubectl get secrets --output=json -n dungla | jq -r '.items[].metadata | select(.name|startswith("vault-token-")).name')
 - TOKEN_REVIEW_JWT=\$(kubectl get secret \$VAULT_HELM_SECRET_NAME --output='go-template={{ .data.token }}' -n dungla | base64 --decode)
 - KUBE_CA_CERT=\$(kubectl config view --raw --minify --flatten --output='jsonpath={.clusters[].cluster.certificate-authority-data}' | base64 --decode)
 - KUBE_HOST=\$(kubectl config view --raw --minify --flatten --output='jsonpath={.clusters[].cluster.server}'))
 - vault write auth/kubernetes/config \
 token_reviewer_jwt="\$TOKEN_REVIEW_JWT" \
 kubernetes_host="\$KUBE_HOST" \
 kubernetes_ca_cert="\$KUBE_CA_CERT"
 - vault write auth/kubernetes/role/devweb-app \
 bound_service_account_names=internal-app \
 bound_service_account_namespaces=dungla \
 policies=devwebapp \
 ttl=24h

- Injecting Secrets With Vault Agents

- Tạo manifest file deployment-vault-injector.yml với Annotation vault-inject=true

```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: web-deployment
  labels:
    app: web
spec:
  replicas: 1
  selector:
    matchLabels:
      app: web
  template:
    metadata:
      labels:
        app: web
      annotations:
        vault.hashicorp.com/agent-inject5: 'true'
        vault.hashicorp.com/role6: 'devweb-app'
        vault.hashicorp.com/agent-pre-populate-only7: 'true'
        vault.hashicorp.com/agent-inject-secret-config8: 'env/my-secret'
    # Environment variable export template
    vault.hashicorp.com/agent-inject-template-config9: |
      {{ with secret "env/my-secret" -}}
      export username="{{ .Data.data.username }}"
      export password="{{ .Data.data.password }}"
      {{- end }}
  spec:
    serviceAccountName: internal-app
    containers:
      - name: web
        image: alpine:latest
        args:
          ['sh', '-c', 'source /vault/secrets/config && echo $username && sleep 100000000']
        ports:
          - containerPort: 9090

```

- Apply manifest file: `kubectl apply -f deployment-vault-injector.yml`

Check log pod ta thấy ở command "echo \$username" sẽ cho ta thấy được giá trị của biến được lấy từ vault server:

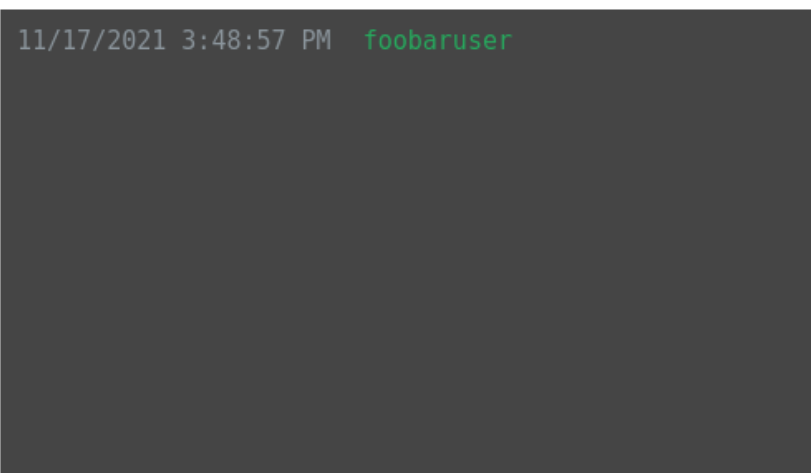
⁵ <http://vault.hashicorp.com/agent-inject>

⁶ <http://vault.hashicorp.com/role>

⁷ <http://vault.hashicorp.com/agent-pre-populate-only>

⁸ <http://vault.hashicorp.com/agent-inject-secret-config>

⁹ <http://vault.hashicorp.com/agent-inject-template-config>

A screenshot of a terminal window with a dark background. At the top left, it shows a timestamp '11/17/2021 3:48:57 PM' in white text, followed by a green username 'foobaruser'. The rest of the terminal is empty.

11/17/2021 3:48:57 PM foobaruser

- Tham khảo:
- <https://devopscube.com/vault-agent-injector-tutorial/>
- <https://www.vaultproject.io/docs/platform/k8s/injector>
- <https://sysdig.com/blog/kubernetes-admission-controllers/>
- <https://kubernetes.io/docs/reference/access-authn-authz/admission-controllers/>
- <https://www.vaultproject.io/docs/platform/k8s/injector/annotations#vault-hashicorp-com-agent-pre-populate-only>
- <https://learn.hashicorp.com/tutorials/vault/kubernetes-sidecar>